

820

**INFORMATION PROCESSOR
SOFTWARE DEVELOPMENT GUIDE**

Xerox Corporation
1341 West Mockingbird Lane
Dallas, Texas 75247

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

WARNING: This equipment generates and uses radio frequency energy and if not installed and used properly, that is, in strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna.
- Relocate the computer with respect to the receiver.
- Move the computer away from the receiver.
- Plug the computer into a different outlet so that computer and receiver are on different branch circuits.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet prepared by the Federal Communications Commission helpful.

"HOW TO IDENTIFY AND RESOLVE RADIO-TV INTERFERENCE PROBLEMS"

This booklet is available from the U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, D.C. 20402, STOCK NO. 004-000-00345-4.

Xerox Corporation reserves the right to make improvements to products without incurring any obligation to incorporate such improvements in products previously sold.

Zilog and Z80 are trademarks of Zilog, Inc., with whom the publisher is not associated.

Xerox® and 820™ are trademarks of Xerox Corporation.

TABLE OF CONTENTS
820
SOFTWARE DEVELOPMENT GUIDE

INTRODUCTION

HARDWARE

CPU BOARD

Microprocessor	2-3
Memory	2-3
Floppy Disk Controller	2-3
CRT Controller	2-3
Parallel Ports	2-3
Serial Ports	2-4
Real Time Clock	2-4
Connector Pin-Outs	2-5
Connector Location	2-8

KEYBOARD

Keyboard Layout	2-9
Keyboard Codes	2-9

POWER SUPPLY

Specifications	2-10
----------------	------

DISK FORMAT

2-11

CRT

Specifications	2-13
----------------	------

SOFTWARE

MONITOR

	3-3
Introduction	3-3
Command Summary	3-3
External Program Interface	3-7

INTERRUPT PROCESSING

	3-9
Storage Location For Mode 2 Interrupt Table	3-9
Device Priority List	3-9

MEMORY MAPPED CRT	
Operational Summary	3-10
Screen Control Codes	3-10
Programming Examples	3-11
Display Character Codes	3-13
SYSTEM PORT NUMBERS	3-14
SERIAL PORTS	3-15
Baud Rate Generator	3-17
PARALLEL PORTS	3-18
TIMER	3-21
REAL TIME CLOCK	3-24
CBIOS MODIFICATION PROCEDURE	3-29
MEMORY ORGANIZATION	
Memory Map	3-31
PROGRAM LISTINGS	
MONITOR ROM VERSION 1.0 (U64 + U63)	4-3
MONITOR ROM VERSION 2.0 (U64)	4-34
MONITOR ROM VERSION 2.0 (U63)	4-59
5.25" CBIOS VERSION 2.0	4-71
8.00" CBIOS VERSION 2.0	4-80
ZILOG DATA	
Z80 CPU	5-3
Z80 PIO	5-25
Z80 CTC	5-37
Z80 SIO	5-49

THEORY OF OPERATION

CENTRAL PROCESSOR	6-3
CRT DISPLAY GENERATOR	6-4
64K RAM AND BANK SWITCHING	6-6
FLOPPY DISK CONTROLLER, SYSTEM PIO AND CTC	6-7
GENERAL PURPOSE PIO AND Z80 SIO	6-8

SCHEMATICS

POWER DISTRIBUTION (ETCH - 1)	7-3
PROCESSOR (ETCH - 1)	7-4
CRT DISPLAY GENERATOR (ETCH - 1)	7-5
RAM (ETCH - 1)	7-6
FLOPPY CONTROLLER, KEYBOARD INPUT, CTC (ETCH - 1)	7-7
GP, PIO, SIO (ETCH - 1)	7-8
POWER DISTRIBUTION (ETCH - 2)	7-9
PROCESSOR (ETCH - 2)	7-10
CRT DISPLAY GENERATOR (ETCH - 2)	7-11
RAM (ETCH - 2)	7-12
FLOPPY CONTROLLER, KEYBOARD INPUT, CTC (ETCH - 2)	7-13
GP, PIO, SIO (ETCH - 2)	7-14

INTRODUCTION

This is the 820 Software Development Guide. This guide contains the information needed to develop programs for the 820 IP and is not intended to teach you how to program.

INTRODUCTION

1 - 2

HARDWARE

HARDWARE

The circuit board that is lying flat under the CRT is the CPU (central processing unit) board. It contains the Z80 microprocessor, the memory and the I/O devices. Reference is made throughout this document to etch 1 and etch 2 CPU boards. The boards can be identified by the following numbers etched on the board: An ETCH 1 board is 140P82629A and an ETCH 2 board is 140P82664A.

MICROPROCESSOR

The microprocessor for the Xerox 820 Information Processor is a Zilog Z80 microprocessor. The processor clock speed is 2.5 Mhz. The Z80 microprocessor is automatically reset at power on or can be manually reset by pressing the reset button on the rear of the display.

MEMORY

The CPU board has 64K of RAM (program memory), 4K of ROM memory (system monitor) and 4K of Ram (CRT memory). The first 16K of system memory can contain either the first 16K of Ram (program memory) or the 4K of ROM memory (system monitor) and 4K of RAM (CRT memory).

When power is applied or the reset switch is depressed the monitor ROM / CRT RAM bank is enabled by hardware and the contents of the monitor ROM are moved by the Z80 microprocessor to the program memory starting at location F000 (hex). When the move is complete the Z80 microprocessor transfers control to location F000 (hex). The only other time that the monitor ROM / CRT RAM bank is enabled is when a character is sent to the screen. When the monitor ROM / CRT RAM bank is enabled the monitor ROM occupies memory at 0000 - 0FFF (hex) and the CRT RAM occupies memory at 3000 - 3FFF (hex). User application programs need not be concerned with the bank switching as it is handled by the monitor and is transparent to transient programs.

FLOPPY DISK CONTROLLER

The CPU board is equipped with a Western Digital 1771-single density floppy disk controller. The clock rate for the 1771 is 2 Mhz. when an 8" disk is connected and 1 Mhz. when a 5.25" disk is connected. When an 8" disk is connected an external data separator is used, when a 5.25" disk is connected the internal data separator is used. This switching is controlled by the signal on the disk interface named 8/N5. It will be a logic 1 when an 8" disk is connected and a logic 0 when a 5.25" disk is connected.

CRT CONTROLLER

The CPU board is equipped with a built in 80 character by 24 line CRT display controller. The refresh memory for the CRT is bank switchable from the systems 64K byte memory space.

The Xerox 820 monitor ROM contains a CRT output driver routine that emulates the characteristics of the Lear Seigler ADM-3A. Many application packages require the terminal type to be specified, if the 820 is not listed as one of the options select the ADM-3A.

PARALLEL PORTS

The Xerox 820 Information Processor has two 8 bit parallel system ports and two 8 bit parallel general purpose ports. The A side of the system Z80 PIO is used for generation of the disc drive select signals, memory bank switching, disc drive identification and disc drive side select. The B side of the system Z80 PIO is used for the parallel keyboard input. The monitor contains an interrupt driven input handler for the keyboard that maintains a 16 character deep FIFO buffer for input data. This makes it possible to do a considerable amount of typing ahead without any characters being lost. If characters are typed while disk access is going on, they may be lost because the disk routines lock out all lower priority interrupts. Any characters received when the FIFO is full will also be lost.

The two general purpose 8 bit ports are unused by the system and can be connected to external parallel devices.

HARDWARE

SERIAL PORTS

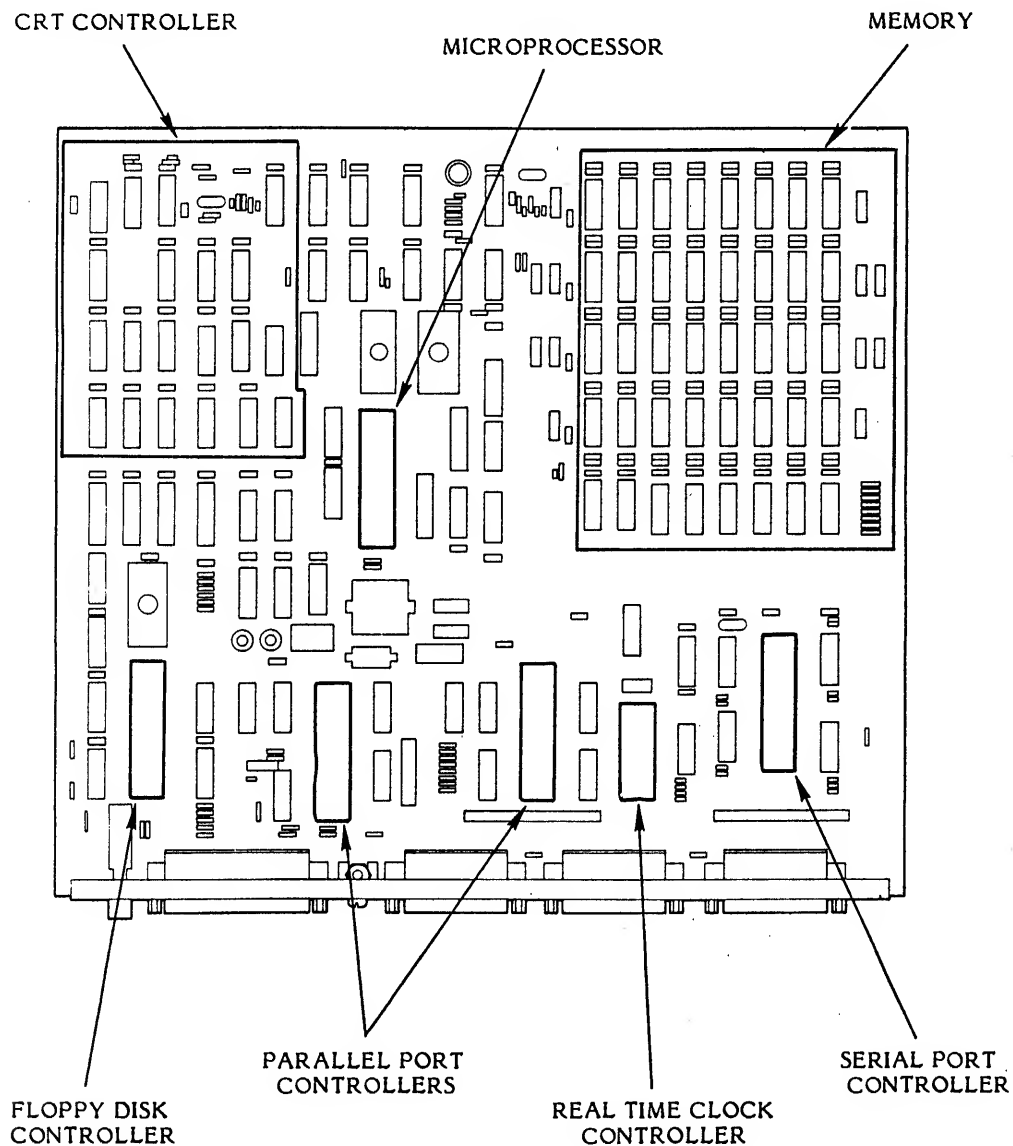
The Z80 SIO supports two full channels of serial I/O with the capability of supporting full RS-232 protocol on both channels. In addition, the A side of the SIO can provide clocks to synchronous modems or receive clocks from the modem.

Channel A of the Z80 SIO can be configured to interface to a modem or a terminal. Refer to the Connector Pin-Outs for J9 and the schematic diagram (sheet 6).

Channel B of the Z80 SIO is dedicated for printer operation and has no strapping options.

REAL TIME CLOCK

The CPU board has a Z80 CTC device that can be used as a timebase for interrupt driven timers, real-time clocks, and other time keeping functions. Channels 2 and 3 are used by the monitor to interrupt the processor once a second. Channel 1 is used by the monitor to perform disk index timing. Channel 0 is not initialized and can be used for other purposes.



HARDWARE

CONNECTOR PIN-OUTS

DISK CONNECTOR

J1	PIN	ASSIGNMENT
	2	8/5% Select
	4	Index
	5	Select 1
	6	Select 2
	7	Side
	8	HDLD
	9	Step In
	10	Step
	11	Write Data
	12	Write
	13	TRK 00
	14	Write Protect
	15	Read Data
	16	Low Current
	17	Ready
	18	+ 12 Volts
	19	+ 5 Volts
	20-37	Ground

KEYBOARD CONNECTOR

J2	PIN	ASSIGNMENT
	1	BIT 0
	2	BIT 1
	3	BIT 2
	4	BIT 3
	5	BIT 4
	6	BIT 5
	7	BIT 6
	8	BIT 7
	9	STROBE
	13	+5 volts
	14-25	Ground

PRINTER CONNECTOR

J3	PIN	ASSIGNMENT
	1	Ground
	2	Receive Data (Input to 820)
	3	Transmit Data (Output from 820)
	4	Clear to Send
	5	Request to Send
	6	Data Set Ready
	7	Ground
	8	Data Terminal Ready
	20	Data Carrier Detect

HARDWARE

MODEM CONNECTOR

J4	PIN	ASSIGNMENT
	1	Ground
	2	Transmit Data
	3	Receive Data
	4	Request to Send
	5	Clear to Send
	6	Data Set Ready
	7	Ground
	8	Carrier Detect
	15	Transmit Clock
	17	Receive Clock
	20	Data Terminal Ready

J5	PIN	ASSIGNMENT
	1	- 12 Volts
	2	+ 12 Volts
	3	+ 12 Volts
	4	Ground
	5	Ground
	6	Ground
	7	+ 12 Volts
	8	+ 5 Volts
	9	+ 5 Volts

J7	PIN	ASSIGNMENT
	3	Vertical Sync
	4	Horizontal Sync
	5	Video
	6-10	Ground

8 BIT GENERAL PURPOSE PARALLEL PORT CONNECTOR

J8	PIN	ASSIGNMENT
	2	port A STROBE
	4	port A READY
	6	port A bit 0
	8	port A bit 1
	10	port A bit 2
	12	port A bit 3
	14	port A bit 4
	16	port A bit 5
	18	port A bit 6
	20	port A bit 7
	22	port B READY
	24	port B STROBE
	26	port B bit 0
	28	port B bit 1
	30	port B bit 2
	32	port B bit 3
	34	port B bit 4
	36	port B bit 5
	38	port B bit 6
	40	port B bit 7
	odd # pins	Ground (ETCH #2 CPU only)

HARDWARE

MODEM PORT OPTION (TERMINAL)

J9	PINS	ASSIGNMENT
	5 6	(M) TXD to Pin 3
	7----8*	(T) TXD to Pin 2
	9 10	(M) RXD from Pin 2
	11----12*	(T) RXD from Pin 3
	13 14	(M) RTS to Pin 5
	15----16*	(T) RTS to Pin 4
	17 18	(M) CTS from Pin 4
	19----20*	(T) CTS from Pin 5
	21 22	(M) DTR to Pin 8
	23----24*	(T) DTR to Pin 20
	25 26	(M) DCD from Pin 20
	27----28*	(T) DCD from Pin 8
	29 30	Clock supplied to Modem as RX Clock
	31----32*	Clock supplied to SIO with RX Clock
	33 34	Modem supplies SIO with RX Clock
	35----36*	Clock supplied to SIO with TX Clock
	37 38	Modem supplies SIO with TX Clock
	39 40	Clock supplied to Modem with TX Clock

* 820 factory settings.

NOTE: (M) Indicates modem (data communications equipment) function.
(T) Indicates terminal (data terminal equipment) function. For instance,
exercising the (T) strap option will allow communication with a modem.
Exercising the (M) strap option would allow communication with a
terminal.

COUNTER/TIMER OPTION (TERMINAL)

J10	PIN		
	System Clock	2 1	CLOCK/TRIGGER 0
	ZC/TO0	4----3*	CLOCK/TRIGGER 1
	ZC/TO1	6 5	CLOCK/TRIGGER 2
	ZC/TO2	8----7*	CLOCK/TRIGGER 3

* 820 factory settings.

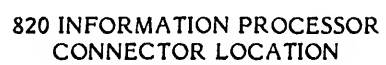
GENERAL PURPOSE PARALLEL PORT OPTION (TERMINAL)

J11	PIN	ASSIGNMENT
	3 4	port B READY polarity
	5 6	port B lower direction
	7 8	port A READY polarity
	9 10	port A upper direction
	11 12	port B upper direction
	13 14	port A STROBE polarity
	15 16	port B STROBE polarity
	17 18	port A lower direction

all odd # pins are grounded

Refer to Parallel Ports in the Software section of this manual for a
description of these jumpers.

HARDWARE



2 - 8

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29	30	31	32		33	34	35	36	37
38	39	40	41	42	43	44	45	46	47	48	49	50		51	52	53	54	55
56	57	58	59	60	61	62	63	64	65	66	67		68	69	70	71	72	
												73	74		75			

KEY NAME	KEY #	UNSHIFTED	SHIFTED	CONTROL
HELP	01	IE	9E	
1	31	21	91	
2	32	40	92	
3	33	23	93	
4	34	24	94	
5	35	25	95	
6	36	5E	96	
7	37	26	97	
8	38	2A	98	
9	39	28	99	
0	10	30	90	
MINUS	11	3D	IF	
EQUAL	12	2D	5F	
BACKSPACE	13	3D	2B	9A
DELETE	14	08	88	
- (PAD)	15	7F	FF	
7 (PAD)	16	2D	AD	
8 (PAD)	17	37	B7	
9 (PAD)	18	38	B8	
TAB	19	39	B9	
Q	20	09	89	
W	21	71	11	
E	22	77	17	
R	23	65	05	
T	24	72	12	
Y	25	74	14	
I	26	79	19	
O	27	75	15	
P	28	69	09	
[29	6F	0F	
]	30	70	10	
ESC	31	5B	1B	
+ (PAD)	32	5D	1D	
4 (PAD)	33	1B	9B	
5 (PAD)	34	2B	AB	
6 (PAD)	35	34	B4	
LOCK	36	35	B5	
	37	36	B6	
	38	---	FUNCTION KEY	

KEY NAME	KEY #	UNSHIFTED	SHIFTED	CONTROL
A	39	61	41	01
S	40	73	53	13
D	41	64	44	04
F	42	66	46	06
G	43	67	47	07
H	44	68	48	08
J	45	6A	4A	0A
K	46	6B	4B	0B
L	47	6C	4C	0C
SEMICOLON	48	3B	3A	7E
APOSTROPHE	49	27	22	60
RETURN	50	0D	0D	8D
LINEFEED	51	0A	0A	8A
UP ARROW	52	01	01	81
1 (PAD)	53	31	31	B1
2 (PAD)	54	32	32	B2
3 (PAD)	55	33	33	B3
C SHIFT	56	---	FUNCTION KEY	---
Z	57	7A	5A	1A
X	58	78	58	18
C	59	63	43	03
V	60	76	56	16
B	61	62	42	02
N	62	6E	4E	0E
M	63	6D	4D	0D
COMMA	64	2C	3C	1C
PERIOD	65	2E	3E	7C
SLASH	66	2F	3F	5C
R. SHIFT	67	---	FUNCTION KEY	---
L. ARROW	68	04	04	84
D. ARROW	69	02	02	82
R. ARROW	70	03	03	83
0 (PAD)	71	30	30	B0
(PAD)	72	2E	2E	AE
L. CTRL	73	---	FUNCTION KEY	---
SPACE BAR	74	20	20	00
R. CTRL	75	---	FUNCTION KEY	---

HARDWARE

POWER SUPPLY

INPUT SPECIFICATIONS

AC Voltage

The power supply is capable of operating from the following voltage and frequency ranges:

90 to 132 volts AC RMS or 198 to 264 volts AC RMS jumper selectable, 47 - 63 HZ

Electrical parameters are specified for 90 to 132 volts AC RMS, 60 HZ operation unless otherwise specified. Output requirements shall be met for the entire input voltage and frequency range.

INPUT CURRENT

The input current will not exceed 2.0 amps RMS. At turn-on, the peak inrush current will not exceed 35 amps at 115V RMS at room temperature of $25 \pm 5^\circ\text{C}$.

INPUT CONNECTION/OUTPUT CONNECTION

PIN NUMBER	SIGNAL NAME
J1 1 3 2	AC Neutral AC Hot Void
P2 1 2 3 4 5 6 7 8 9	-12VDC +12VDC #1 +12VDC #1 DC Ground DC Ground DC Ground +12VDC #2 +5VDC +5VDC

OUTPUT SPECIFICATIONS

OUTPUT DC VOLTS	MIN. LOAD CURRENT	CONTINUOUS LOAD CURRENT MAXIMUM	PEAK LOAD CURRENT MAXIMUM	RIPPLE P-P MV MAX.	TOLERANCE % MAXIMUM
+5	2.0	4.65	4.65	50	± 2
#1 + 12	0.50	1.80	2.8	50	± 5
- 12	0.25	0.50	0.5	50	± 5
#2 + 12	0.50	2.0	2.0	+50	± 5

HARDWARE

Over Voltage Protection

The +5.0VDC output shall be overvoltage protected. The over voltage protection circuitry shall be set to operate when the voltage output is between 120 and 140% of rated voltage.

Fuse Replacement

F1 (2.5 amp normal blow)

DISK FORMAT

The XEROX 820 Information Processor is equipped with two (2) Shugart SA400L (5 $\frac{1}{4}$ ") drives, two Shugart SA800 (8") drives, or two Shugart SA450 (5 $\frac{1}{4}$ ") drives.

A format is divided into three (3) parts, field A, field B, and field C. Field A is written at the start of each track known as the preamble. Field B is written once for each sector which consists of a gap between sectors, ID fields, and a data field. Field C is written at the end of each track and is known as a postamble.

The XEROX 820 Information Processor disks are initialized in the following formats:

PARAMETER	8"SSSD	5 $\frac{1}{4}$ "SSSD	5 $\frac{1}{4}$ "DSSD
Tracks	77	40	40
Sectors	26	18	18
Bytes/Sector	128	128	128
# of Reserved Track for OS	2	3	3
Disk Capacity	241K	81K	172
Sides	1	1	2

5 $\frac{1}{4}$ " Format

	Number of Bytes	Hex Value of Bytes	Comment
Field A -	16	FF	Preamble on Gap 4A
	4	00	Gap 3
	1	FE	ID Address Mark
	1	XX	Track #
	1	00	
	1	XX	Sector #
	1	00	
*Field B -	1	F7	Generate CRC
	11	FF	Gap 2
	6	00	
	1	FB	Data Address Mark
	128	E5	Data Field 'E5' Data
	1	F7	Generate CRC
	8	FF	Gap 8
Field C -	101	FF	Postamble Gap 4B

* Repeated for number of sectors per track.

HARDWARE

DISK FORMAT (continued)

8" Format

	Number of Bytes	Hex Value of Bytes	Comment
Field A -	28	FF	Preamble - Write at the
	6	00	start of each track
	1	FC	
	26	FF	
	6	00	Gap 3
	1	FE	ID Address Mark
	1	XX	Track #
	1	00	
	1	XX	Sector #
	1	00	
*Field B -	1	F7	Generate CRC
	11	FF	Gap 2
	6	00	
	1	FB	Data Address Mark
	128	E5	Data Field '5' Data
	1	F7	Generate CRC
	27	FF	Gap 3
Field C -	247	FF	Postamble Gap 4B

* Repeated for number of sectors per track.

CRT

SPECIFICATIONS

Power

The CRT monitor shall function within the limits specified herein when the following power is supplied.

Voltage: $+12.0 \pm 5.0\%$ VDC at 2.0 A DC maximum.
Ripple: 50 MV P-P synchronous or nonsynchronous with refresh or power frequency.

Phosphor

TYPE

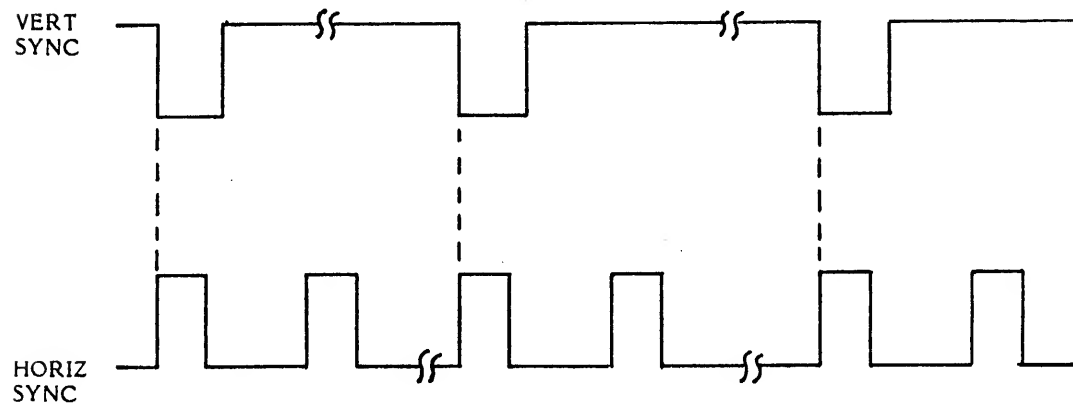
Aluminized	P4
Fluorescence	White (W)
Phosphorescence	White (W)
Persistence	Short

Resolution

With a 240 active line raster adjusted to 8.5 X 5.3 inches usable area and a brightness level of 37 ± 2 foot-lamberts (bright screen - no characters), the resolution shall be as specified below. This specification shall be verified by supplying a synchronized video square wave signal to the unit and viewing the resultant screen image. Waveform duty cycle shall be $0.5 \pm 10\%$. Signal frequency shall be 8.12 MHz minimum. Individual black or white bars shall be visible with the unaided eye at a distance of 12 inches from the CRT faceplate. Optical magnification may be used only for dimensional and quantitative measurements.

Resolution at centers (within 1" dia. circle) - 100 lines/in min.

SIGNAL TIMING



HARDWARE

Video bit rate (time)	10.694 MBPS	(93.51 nS)
Active bits per horizontal line (time)	560	(52.366 uS)
Horizontal line blanking bits (time)	140	(13.091 uS)
Horizontal front porch-bits (time)	0	
Horizontal sync pulse-bits (time)	140	(13.091 uS)
Horizontal back porch-bits (time)	0	
Total bits per line (time)	700	(65.457 uS)
Horizontal rate	15.278 KHZ	
Active lines per field (time)	240	(15.710 mS)
Vertical blanking lines (time)	20	(1.309 mS)
Vertical front porch-bits (time)	0	
Vertical sync pulse-bits (time)	20	(1.309 mS)
Vertical back porch-bits (time)	0	
Vertical retrace (lines)	8 TYP.	
Total lines per field	260	
Field rate (time)	58.758 Hz	(17.019 mS)

Input Signal Description

Parameter	Video	Horizontal Sync	Vertical Sync	Brite
Input Type	Single Ended	Single Ended	Single Ended	—
Z In	R _{Shunt} 150 ± 5% C _{Shunt} 30 pf max	R _{Shunt} = 2K Minimum C _{Shunt} = 50 pf max		— —
Amplitude	Low = 0 = 0 to + 0.4V High = 1 = 2.5 ± 0.1V	Low = 0 = 0 to 0.4V High = 1 = 2.0 to 5.0V		300V Max
Polarity	1 = Brite 0 = Dark	1 = Sync	0 = Sync	—
Rate	10.69 Mbps Max	15,278 Hz ± 100Hz		DC
Rise/Fall Times 10% to 90%	Less than 20 nsec	Less than 100 nsec		—

HARDWARE

SOFTWARE

SOFTWARE

MONITOR

INTRODUCTION

The XEROX 820 system monitor is the basic control program for the single-board computer. It begins execution when the computer is first turned on, or whenever the reset button is pressed, and resides in the top 4K of RAM memory (F000-FFFF).

The monitor provides two essential functions for the system. It is the initial software level of the computer and it contains the routines that initialize and control all the basic system input/output resources. The "front panel" functions of the monitor include commands to display and alter the contents of memory and I/O ports, to begin execution at a given address, enter typewriter mode, and to bootstrap programs from disk. The basic I/O functions of monitor provide driving routines for the built-in CRT display and keyboard input, and the floppy disk controller. In this capacity the monitor is always active, even when application programs like the CP/M disk operating system have control of the CPU.

The following sections of this manual will explain how to use the console monitor commands, what facilities are provided by the resident I/O handlers, and how to interface applications programs to the monitor.

COMMAND SUMMARY

The Xerox 820 monitor enters the command mode after it has initialized the system following a power-on or a reset. The following sign-on message is displayed on the console output device as an indication that the monitor is ready to accept commands.

```
... XEROX 820 ...  
Enter A for BOOT  
Enter T for TYPEWRITER  
*  
_
```

Commands consist of a single character command name and zero to three hexadecimal numeric parameters separated by commas or spaces. The command line may be entered using upper case or lower case letters. A carriage return is used as the terminator. Errors within a line can be corrected by typing backspace to delete the last character. If a line is entered with an unknown command name, an invalid number or parameters or an out-of-range parameter, an error message will be displayed and the command will not be executed.

The user may wish to halt long running commands like the memory dump before they are finished. This can be done by typing carriage return while the command is doing output. Output can also be frozen temporarily and then re-started by typing repeatedly on the space bar.

The following table summarizes the monitor's command set. The items enclosed in angle brackets represent the numeric parameters expected by the command. A detailed description of each command is provided in the following pages.

<u>Command</u>	<u>Format</u>
d(ump)	... D (start), (end)
m(emory)	... M (address)
x(test)	... X (start), (end)
f(ill)	... F (start), (end), (constant)
c(opy)	... C (source_start), (source_end), (dest_start)
g(oto)	... G (address)
r(ead)	... R (unit), (track), (sector)
a(boot)	... A(boot)
t(ypewriter)	... T
i(nput)	... I (port)
o(utput)	... O (port), (data)

NOTE: All of the Monitor parameters are in hexadecimal.

SOFTWARE

DUMP COMMAND (D)

The dump command outputs a tabular display of the contents of memory in hexadecimal and ASCII representation. Each display line has the following format:

```
aaaa dd dd dd dd dd dd dd dd dd dd dd dd dd dd dd cccccccccccccc
```

where aaaa is the starting memory address of the line in hexadecimal, the dd's are the hex values of the 16 bytes of data starting at location aaaa, and the c's are the ASCII characters equivalent to each data byte. Bytes less than 20 hex are replaced in the ASCII portion of the dump by period.

The dump command accepts zero, one or two address parameters. If two addresses are specified, the block of memory between those two locations will be displayed. Entering only one address will display 256 bytes of memory starting at the specified location. Typing 'D' with no parameters will cause the routine to display the 256 byte block of memory starting at the last address displayed by the dump command.

MEMORY COMMAND (M)

The memory examine/change command allows the contents of individual memory locations to be read from and written into using the monitor. This command accepts one parameter representing the memory address at which to begin examining data. The display format is as follows:

```
AAAA DD _
```

where AAAA is the current memory address and DD is the hexadecimal value of the data in that location. After displaying the contents of a memory location, the routine waits for one of the following items to be input from the console.

- Typing a carriage return will cause the routine to display the data at the next memory location, with no modification of content.
- Typing a minus sign will have a similar effect, except the address is decremented instead of incremented.
- Typing a two digit hexadecimal number will cause that number to be stored at the displayed address. The new data is stored as soon as the second digit is entered, with no terminating character required.
- Typing any character other than carriage return, a minus sign or a hexadecimal digit will cause the command to terminate.

TEST COMMAND (X)

This command allows the user to test memory for errors. Any portion of memory may be tested except the area reserved for the monitor (F000 to FFFF hex). Two parameters are required from the user; the starting address and ending address of the memory block to be tested. Only the high order 8 bits of the addresses entered are actually used. If no errors occur, the test routine will output a plus sign every time a test pass is done. A total of 256 plus signs must be output for all possible test patterns to have been tried. When errors are detected an error line will be output in the following format:

```
AAAA DD should=XX
```

where AAAA is the address of a location that fails to test, DD is the data read back from the location, and XX is the test pattern that was written there.

FILL COMMAND (F)

The fill command allows blocks of memory to be filled with a fixed data constant. Three parameters are required in the command line; a starting memory address, an ending address and a fill constant. Each location in the specified block of memory has the constant written into it and then read back again to check for memory errors. An error line like the one described for the 'X' command is printed for any locations that fail to verify.

SOFTWARE

COPY COMMAND (C)

The copy command allows blocks of data to be moved around in memory. Three parameters are required in the command line; a starting memory address, an ending address, and a destination address. The contents of the block of memory bounded by the first two addresses is copied to the block starting at the third address. As with the fill command, a test is made to verify that each byte of the destination block, when read back, is the same as the corresponding byte in source block.

GO TO COMMAND (G)

The goto command allows control of the CPU to be passed to another program by the monitor. This command requires a single parameter from the user representing the address at which to begin execution. The monitor actually passes control to the specified location by executing a CALL instruction. This makes it possible for the external routine to return to the monitor by doing a RET, assuming it does not re-load the stack pointer and loose the return address to the monitor.

READ COMMAND (R)

The read command allows individual disk sectors to be read into memory and displayed on the console. Three parameters are required; a drive unit number (range 0 to 1), a track number (range 0 to 27 for 5.25" disks or range 0 to 4D for 8" disks) and a sector number (range 1 to 12 for 5.25" disks or range 1 to 1A for 8" disks). The command routine performs a drive select, track seek and sector read sequence using the supplied parameters. If no errors occur, the contents of the input buffer will be dumped out the 'D' command format. In the event of a disk error, a diagnostic message will be printed in the following format:

disk error XX UAA TBB SCC

where XX represents the 1771 disk controller error status code, AA is the unit number, BB is the track number, and CC is the sector number. The error code is composed of eight bits of status information as described in the table below:

<u>bit</u>	<u>read/write</u>	<u>seek/restore/select</u>
7	drive not ready	drive not ready
6	write protected	unused
5	write fault	unused
4	record not found	seek error
3	crc error	crc error
2	lost data	cannot restore
1	unused	unused
0	always=1	always=0

The least significant bit (LSB) of the error code indicates which of the above sets of error conditions is applicable. If the LSB=1 the disk error was generated by a read or write operation, otherwise it was caused by a seek, restore, or select operation.

BOOT COMMAND (A)

The boot command command is used to load and begin execution of a one sector long bootstrap loader from drive unit zero. The most common use of this command will be to boot up the CP/M disk operating system, although it is not necessarily restricted to this purpose only.

The boot works by reading the contents of track 0, sector 1 into memory at location 80 hex and the jumping to that address to start execution of the code just read in. Normally the routine on sector 1 will be a small loader that in turn reads in a larger program such as the operating system.

TYPEWRITER COMMAND (T)

This command allows the XEROX 820 to be used as a standard electronic typewriter. All key strokes will be typed directly on the 630 printer in a direct print mode, without displaying any typed information on the screen.

INPUT COMMAND (I)

This command allows the contents of input ports to be read from using the monitor. It operates very much like the memory examine command, except that input ports are being examined instead of memory locations. A single parameter representing a port number is expected in the command line. The contents of adjacent ports can then be examined by typing carriage return or a minus sign as in the 'M' command. Typing any other key will cause the routine to terminate.

OUTPUT COMMAND (O)

The output command is provided to allow output ports to be written to using the monitor. Two parameters are expected in the command line; a port number and a data byte to be output to that port. Both parameters should be between 0 and FF hex. After outputting the specified data to the port, this routine simply returns to the monitor instead of stepping to the next location like the input command. This makes it possible to use the output command to initialize Z-80 peripheral devices like the SIO, PIO and CTC.

EXTERNAL PROGRAM INTERFACE

This section gives the locations and calling sequences of the user accessible I/O routines in the XEROX 820 monitor.

XEROX 820 subroutines are accessed via a table of JUMP instructions beginning at memory location F000 hex. All monitor calls should be made to these entry points, since the actual addresses of the routines inside XEROX 820 will vary between different releases. Parameter passing conventions for the monitor fall into one of two groups. The character oriented I/O routines all pass data using the A register, while the disk routines pass parameters in C and HL and return status information in A.

Storage for the monitor's stack and working variable occupies the top 256 bytes of memory, from FF00 to FFFF hex. The mode 2 interrupt vector table takes up the first 32 bytes of this block.

XEROX 820 SUBROUTINE ENTRY POINTS

<u>LOCATION</u>	<u>FUNCTION</u>	<u>PARAMETERS</u>	<u>DESCRIPTION</u>
F000	INIT . . .	IN: none OUT:does not return	Perform cold start initialization of XEROX 820 monitor and enter command mode.
F003	PROMPT . . .	IN: none OUT:does not return	Enter XEROX 820 monitor command mode with no initialization
F006	CONST . . .	IN: none OUT:status in A	Test for data ready in console input FIFO and return status in A. If data is available then A=FF hex, else A=00.
F009	CONIN . . .	IN: none OUT:character in A	Return character from console input FIFO in A. If FIFO is empty then loop until character is input.
F00C F00F	CRTOUT..	IN: character in A OUT:none	Output character passed in A to the memory-mapped CRT display.
F012	SIOST..	IN: none OUT:status in A	Test for received data available from SIO channel B and return status in A. If data is available then A=FF hex, else A=00.
F015	SIOIN . . .	IN: none OUT:character in A	Return received data from SIO channel B in A. Loop until data is received if none is available on entry.
*F018	SIOOUT . . .	IN: character in A OUT:none	Output charater passed in A to SIO channel B transmit register.
F01B	SELECT . . . **	IN: unit number in C OUT:status in A	Select specified drive for future restore, seek, read or write command. If the drive is not ready, then the currently selected drive is left on.
F01E	HOME . . . **	IN: none OUT:status in A	Move read/write head to home position at track 0 and verify if it got there.
F021	SEEK . . . **	IN: track number in C OUT:status in A	Move read/write head to specified track and verify if it got there.

SOFTWARE

EXTERNAL PROGRAM INTERACE (continued)

<u>LOCATION</u>	<u>FUNCTION</u>	<u>PARAMETERS</u>	<u>DESCRIPTION</u>
F024	READ . . .	IN: sector number in C buffer pointer in HL ** OUT:status in A	Read specified sector on current track into memory data buffer.
F027	WRITE . . .	IN: sector number in C buffer pointer in HL ** OUT:status in A	Write specified sector on current track from memory data buffer.

* Inoperative on level 2.0 ROM

** If the status returned in the A register is 00 the function was performed with no errors. Error conditions returned a the A register are as follows:

<u>bit</u>	<u>read/write</u>	<u>seek/restore/select</u>
7	drive not ready	drive not ready
6	write protected	unused
5	write fault	unused
4	record not found	seek error
3	crc error	crc error
2	lost data	cannot restore
1	unused	unused
0	always=1	always=0

The least significant bit (LSB) of the error code indicates which of the above sets of error conditions is applicable. If the LSB=1 the disk error was generated by a read or write operation, otherwise it was caused by a seek, restore, or select operation.

SOFTWARE

INTERRUPT PROCESSING

The XEROX 820 monitor takes advantage of the powerful interrupt handling capabilities of the Z80 microprocessor. Interrupts are utilized in the I/O drivers for the console keyboard input, the real-time clock and the floppy disk controller. All necessary initialization tasks and interrupt service routines for these devices are contained in the monitor.

For the most part, the operation of the interrupt mechanism should be transparent to applications programs that will run on the XEROX 820. A few precautions must be taken however, to insure that user written software does not adversely effect the operation of the system. The following list describes the major hazards to the interrupt system;

Interrupts should not be disabled permanently by user code, as this will lock-up the console input and real-time-clock routines.

The Z80 'I' register should never be altered.

The CPU operates in Z80 interrupt mode 2 and should not be switched to either of the other two interrupt modes.

Adequate stack space must be reserved in user programs to allow at least one level of stack for interrupt return addresses.

The monitor initializes the Z80 'I' register to point to the system interrupt vector table at location FF00 to FF1F hex. This table contains pre-assigned vector locations for all the peripheral devices on the XEROX 820.

STORAGE ALLOCATION FOR MODE 2 INTERRUPT TABLE

FF00	SIOVO:	DEFS2	;Z80 SIO port B xmit buffer empty
FF02	SIOV1:	DEFS2	;Z80 SIO port B external/status change
FF04	SIOV2:	DEFS2	;Z80 SIO port B receive data available
FF06	SIOV3:	DEFS2	;Z80 SIO port B special receive condition
FF08	SIOV4:	DEFS2	;Z80 SIO port A xmit buffer empty
FF0A	SIOV5:	DEFS2	;Z80 SIO port A external/status change
FF0C	SIOV6:	DEFS2	;Z80 SIO port A receive data available
FF0E	SIOV7:	DEFS2	;Z80 SIO port A special receive condition
FF10	CTCVO:	DEFS2	;Z80 CTC channel 0 interrupt
FF12*	CTCV1:	DEFS2	;Z80 CTC channel 1 interrupt
FF14*	CTCV2:	DEFS2	;Z80 CTC channel 2 interrupt
FF16*	CTCV3:	DEFS2	;Z80 CTC channel 3 interrupt
FF18	SYSVA:	DEFS2	;system Z80 PIO port A interrupt
FF1A*	SYSVB:	DEFS2	;system Z80 PIO port B interrupt
FF1C	GENVA:	DEFS2	;general purpose Z80 PIO port A interrupt
FF1E	GENVB:	DEFS2	;general purpose Z80 PIO port B interrupt

* Vectors used by the Monitor ROM (Version 1.0 & 2.0)

DEVICE PRIORITY LIST

The Interrupt Priority chain is organized high to low as follows:

Z80 SIO CHANNEL A
Z80 SIO CHANNEL B
SYSTEM Z80 PIO PORT A
SYSTEM Z80 PIO PORT B
GENERAL PURPOSE Z80 PIO PORT A
GENERAL PURPOSE Z80 PIO PORT B
Z80 CTC CHANNEL 0
Z80 CTC CHANNEL 1
Z80 CTC CHANNEL 2
Z80 CTC CHANNEL 3

SOFTWARE

MEMORY MAPPED CRT

CRT DRIVER OPERATIONAL SUMMARY

All character codes between 32 (20 hex) and 127 (7F hex) are directly displayable on the screen.

All character codes between 00 and 31 (1F hex) are interpreted as control characters. Only 12 of these codes have an effect on the CRT display, and are described in the table below. The remaining 20 are treated as nulls.

New characters are stored on the screen at the location occupied by the cursor. The cursor is then moved one space to the right.

If the cursor is positioned at a screen location occupied by a non-blank character, the presence of the cursor will be indicated by making the overlaid character blink.

If a linefeed (LF) is output when the cursor is on the bottom line of the screen, the entire display is scrolled up one line and a new blank line is created on the bottom.

If the displayed character is output when the cursor is in the right most column of the screen, an automatic carriage return and linefeed is generated.

820 SCREEN CONTROL CODES

DECIMAL CODE	HEX CODE	ASCII NAME	CRT-EFFECT
08	08	BS	Cursor Left (backspace)
09	09	HT	Horizontal Tab
10	0A	LF	Cursor Down (linefeed)
11	0B	VT	Cursor Up
12	0C	FF	Cursor Right
13	0D	CR	Carriage Return
17	11	DC1	Clear to end of screen
24	18	CAN	Clear to end of line
26	1A	SUB	Clear screen
27	1B	ESC	Initiate escape sequence
30	1E	RS	Home cursor
31	1F	VS	Display special character

SOFTWARE

PROGRAMMING EXAMPLES

Cursor Left

Moves the cursor to the left one column. If the cursor is in the left most column of the screen, this character has no effect.

Example in Basic to move the cursor one space to the left:

```
100 PRINT CHR$(8);  
110 END
```

Horizontal Tab

Moves the cursor right to the next tab stop. The tab stops are fixed at every eighth column, starting from the left.

Example in Basic to move the cursor to the right 3 tab stops:

```
100 FOR X = 1 TO 3  
120 PRINT CHR$(9);  
130 NEXT X  
140 END
```

Cursor Down (linefeed)

Moves the cursor down one line on the screen. If the cursor is at the bottom most line, the screen is scrolled up and a blank line is created on the bottom. The top line is lost.

Example in Basic to move the cursor down 5 lines:

```
100 FOR X = 1 to 5  
110 PRINT CHR$(10);  
120 NEXT X  
130 END
```

Cursor Up

Moves the cursor up one line on the screen. If the cursor is on the top of the screen it rolls around to the bottom.

Example in basic to move the cursor up 5 lines:

```
100 FOR X = 1 to 5  
110 PRINT CHR$(11);  
120 NEXT X  
130 END
```

Cursor Right

Moves the cursor to the next column to the right. If the cursor is in the right most column, there is no effect.

Example in Basic to move the cursor 5 spaces to the right:

```
100 FOR X = 1 to 5  
110 PRINT CHR$(12);  
120 NEXT X  
130 END
```

Carrier Return

Moves the cursor to the left most column of the screen.

Example in Basic to move the cursor to the left column:

```
100 PRINT CHR$(13);  
110 END
```

SOFTWARE

Clear to End of Screen

Clears the contents of the screen from the current cursor position to the end of the bottom line.

Example in Basic to Clear to the end of the screen:

```
100 PRINT CHR$(17);
110 END
```

Clear to End of Line

Clears the contents of the line the cursor is on, from the cursor position to the end of the line.

Example in Basic to Clear to the end of the line:

```
100 PRINT CHR$(24);
110 END
```

Clear Screen

Clears the entire screen regardless of the current cursor position and places the cursor in the top left corner of the screen.

Example in Basic to clear the screen:

```
100 PRINT CHR$(26);
```

Escape Sequence

Used to initiate an XY cursor positioning sequence. The cursor can be moved to an arbitrary location on the screen by outputting a 4 character sequence composed of: 1) ESCAPE - CHR\$(27), 2) EQUALS sign - CHR\$(61), 3) ROW # (0-23) + 32, 4) COLUMN # (0-79) + 32.

Example in Basic to clear the screen and position the cursor on Row 10, Column 40 and print an X.

```
100 PRINT CHR$(26);
120 PRINT CHR$(27);CHR$(61);CHR$(10+32);CHR$(40+32);
130 PRINT 'X';
140 END
```

Home Cursor

Moves the cursor to the top left corner of the screen, without altering any characters on the display.

Example in Basic to home the cursor:

```
100 PRINT CHR$(30);
110 END
```

Display Special Character

Functions as a prefix character to force the output of special symbols in the character generator. This character must precede any character in the display code chart from 00 thru 1F (hex).

For example, to display the vertical bar character (code 19 hex on the display code chart), the following basic program could be used:

```
100 PRINT CHR$(31);
110 PRINT CHR$(25);
120 END
```

DISPLAY CHARACTER CODES

This table shows the code for each character to be displayed by the XEROX 820. Each character is defined by a unique eight bit code which is represented by a hexadecimal code 'XY' where X represents the 4 most significant bits of the code and Y represents the 4 least significant bits of the code.

There are a total of 128 characters in the font set. Therefore, Y represents a hexadecimal number from 0 to F, and X represents a hexadecimal number from 0 to 7. Therefore, the complete font set is defined by codes from 00 to 7F.

If the most significant bit of the eight bit code is set to '1', then the complete font set is duplicated with the blink attribute set. The blinking set of characters is then defined by codes from 80 to FF (Level 2.0 ROM only).

	Y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
X																	
0		□	¢	■		§	½	¼	±	←	↑	↓	→	↔	⊗	⊙	↔
1		³	²	°	—	↓	¶	±	μ	†	‡	↙	↘	⊕	↻	↻	↻
2			!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3		0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4		@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5		P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6		`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7		p	q	r	s	t	u	v	w	x	y	z	{		}	~	™

SYSTEM PORT NUMBERS

PORT 00 = CHANNEL A BAUD RATE (WRITE ONLY)

PORT 04 = Z80 SIO CHANNEL A DATA

PORT 06 = Z80 SIO CHANNEL A CONTROL

PORT 0C = CHANNEL B BAUD RATE (WRITE ONLY)

PORT 05 = Z80 SIO CHANNEL B DATA

PORT 07 = Z80 SIO CHANNEL B CONTROL

PORT 08 = GENERAL PURPOSE Z80 PIO PORT A DATA

PORT 09 = GENERAL PURPOSE Z80 PIO PORT A CONTROL

PORT 0A = GENERAL PURPOSE Z80 PIO PORT B DATA

PORT 0B = GENERAL PURPOSE Z80 PIO PORT B CONTROL

PORT 10 = 1771 STATUS/COMMAND REGISTER

PORT 11 = 1771 TRACK REGISTER

PORT 12 = 1771 SECTOR REGISTER

PORT 13 = 1771 DATA REGISTER

PORT 14 = CRT SCROLL REGISTER (WRITE ONLY)

PORT 18 = Z80 CTC CHANNEL 0

PORT 19 = Z80 CTC CHANNEL 1

PORT 1A = Z80 CTC CHANNEL 2

PORT 1B = Z80 CTC CHANNEL 3

PORT 1C = SYSTEM Z80 PIO PORT A DATA

PORT 1D = SYSTEM Z80 PIO PORT A CONTROL

PORT 1E = SYSTEM Z80 PIO PORT B DATA (KEYBOARD)

PORT 1F = SYSTEM Z80 PIO PORT B CONTROL (KEYBOARD)

SOFTWARE

SERIAL PORTS

A Z80 SIO provides the 820 with a serial interface to the outside world. The Z80 SIO has two Channels, A & B. The printer port is Channel B and the modem port is Channel A. Channel B is initialized by the ROM MONITOR, Channel A is uninitialized.

The monitor initializes Channel B as follows:

SIO-Register *	DATA (Hex)	COMMENTS
4	45	16X Clock, 1 Stop Bit, Odd Parity
1	04	Status affects Vector
3	41	RX-7 Bits/Character, Rx-enable
5	2A	TX-7 Bits/Character, Tx-enable, RTS
2	00	Base Interrupt Vector

Channel B Baud Rate is set to 300 baud by the monitor, and to 1200 baud when the CP/M disk is loaded.

The software supplied by XEROX uses ETX/ACK protocol to "handshake" with the printer. This handshaking is done in the CBIOS and can be changed to use other methods of handshaking. Let's assume that we have a serial printer that has a Logic TRUE (high) on pin 20 when it is ready to receive a character. When it cannot receive another character Pin 20 will be low for busy. The software to accomplish this follows:

```

;
;List device output routine, assume character is in the C register.
;Handshake with the printer using Pin 20 on the interface.
;
;Note: Pin 20 is connected to the  $\overline{\text{DCD}}$  pin on the SIO.
;
;Constants - For Z80 SIO Channel B
;
SIOBCO EQU 07 ;SIO Channel B Control
SIOBDA EQU 05 ;SIO Channel B Data Port
SIORES EQU 10H ;SIO Reset External Status Command
RDYMSK EQU 00001100B ;Mask to check for SIO and Printer Ready
;
;
LSTOUT: LD A,SIORES ;Get External Reset Command to A Register
        OUT (SIOBCO),A ;Send to Channel B Control Port
        IN A,(SIOBCO) ;Read Channel B Control Port
        AND RDYMSK ;Mask of Everything of Interest
        CP RDYMSK ;Check for Expected Result
        JR NZ,LSTOUT ;Repeat until Everything is Ready
        LD A,C ;Get Character to A Register
        OUT (SIOBDA),A ;Send to Data Port
        RET ;Return to Caller

```

CHANNEL A INITIALIZATION

Channel A is not initialized, before using Channel A you should set-up the desired operating mode. As an example, the following sub-routine could be used to initialize Channel A.

```

;
;Channel A Z80 SIO Initialization Routine
;
;
CONSTANTS FOR SIO Channel A
;
SIOACO EQU 06 ;Channel A Z80 SIO Control Port
SIOADA EQU 04 ;Channel A Z80 SIO Data Port
BAUDA EQU 00 ;Channel A Baud Rate Port
XMTRDY EQU 00000100B ;Transmit Buffer Ready Bit
RCVRDY EQU 00000001B ;Receive Character Ready Bit

        LD C,SIOACO ;Get Port Number to C Register

```

SOFTWARE

```

        LD    B,6           ;Byte Count to Register B
        LD    HL,STABL      ;Point H & L Register to the Start of the table
        OTIR                ;Do output and Increment
OVRTBL: LD    A,05          ;Set A Register for 300 Baud
        OUT   (BAUDA),A     ;Set Channel A Baud Rate
        RET

STABL:  DEFB 04             ;Select Register #4
        DEFB 01000100B      ;16X Clock, 1 Stop Bit, No Parity
        DEFB 03             ;Select Register #3
        DEFB 01000001B      ;7 Bits/RX Character, RX-enable
        DEFB 05             ;Select Register #5
        DEFB 10101010B      ;7 Bits/TX Character, DTR active, TX-enable

```

The following routines will do input, output and status checking on Channel A:

```

;
;CHAOUT - Subroutine to output the charater in the C Register to Channel A
;

```

```

CHAOUT: IN    A,(SIOACO)    ;Read Channel A Control Port
        AND   XMTRDY        ;Check X-Mit Buffer Empty Flag
        JR    Z,CHAOUT      ;Repeat until Ready
        LD    A,C           ;Character to A Register
        OUT   (SIOADA),A    ;Output Character
        RET                ;Back to Caller

```

```

;CHASTA - Subroutine to check the receive status of Channel A
;
;           A Register = 00 if no character is ready
;           A Register = FF if a character is ready
;

```

```

CHASTA: IN    A,(SIOACO)    ;Read Channel A Control Port
        AND   RCVRDY        ;Check Receive Character Available
        RET   Z             ;If Zero Return
        LD    A,0FFH        ;Put FF in A Register
        RET                ;Back to Caller

```

```

;CHAINP - Subroutine to read a character from Channel A
;
;           Return with the character in the A Register
;

```

```

CHAINP: CALL  CHASTA        ;Get Receive Status
        JR    Z,CHAINP      ;Repeat until Character is r
        IN    A,(SIOADA)    ;Get Character to A Register
        RET                ;Back to Caller

```

SOFTWARE

BAUD RATE GENERATOR

The 820 provides the user with two programmable baud rate generators. Channel A baud rate resides at port 00 hex and is write only. Channel B baud rate resides at port 0C hex and is also write only. The programming procedure is as follows:

Load the accumulator with the hex value for the desired BAUD rate (See table below). Output the contents the accumulator to the desired serial channel.

The following sub-routine would initialize Channel A for 9600 Baud and Channel B for 300 Baud.

```
LD    A,0EH           ;Code for 9600 Baud to A Register
OUT   (0),A           ;Output to Channel A
LD    A,05             ;Code for 300 Baud to A Register
OUT   (0CH),A         ;Output to Channel B
RET
```

BAUD RATE TABLE

00 hex =	50 Baud
01 hex =	75 Baud
02 hex =	110 Baud
03 hex =	134.5 Baud
04 hex =	150 Baud
05 hex =	300 Baud
06 hex =	600 Baud
07 hex =	1200 Baud
08 hex =	1800 Baud
09 hex =	2000 Baud
0A hex =	2400 Baud
0B hex =	3600 Baud
0C hex =	4800 Baud
0D hex =	7200 Baud
0E hex =	9600 Baud
0F hex =	19.2 Kbaud

PARALLEL PORTS

The 820 has two Z80 PIO's on the CPU Board, one is dedicated for the systems' use, the other is available to the user and is called the General Purpose (GP) PIO. The Port assignments for the GP PIO are as follows:

<u>PORT #</u>		<u>Description</u>
08	---	GP PIO PORT A DATA
09	---	GP PIO PORT A CONTROL
0A	---	GP PIO PORT B DATA
0B	---	GP PIO PORT B CONTROL

Description of hardware jumpering options on the GP-PIO (J11):

<u>J11</u>	<u>PINS</u>	<u>DESCRIPTION</u>
	9-10	Port A, Bit 7 through Bit 4 Direction Control ON - Outputs from the 820 OFF - Inputs to the 820
	17-18	Port A, Bit 3 through Bit 0 Direction Control ON - Outputs from the 820 OFF - Inputs to the 820
	7-8	ARDY Pulse (PORT A) ON - Non-inverted OFF - Inverted
	13-14	ASTB Pulse (PORT A) ON - Non-inverted OFF - Inverted
	11-12	PORT B, Bit 7 through Bit 4 Direction Control ON - Outputs from the 820 OFF - Inputs to the 820
	5-6	PORT B, Bit 3 through Bit 0 Direction Control ON - Outputs from the 820 OFF - Inputs to the 820
	3-4	BRDY Pulse (PORT B) ON - Non-inverted OFF - Inverted
	15-16	BSTB Pulse (PORT B) ON - Non-inverted OFF - Inverted

The hardware jumpering on J11 determines the direction select of the transceiver (74LS243) that is connected between the Z80 PIO and J-8. The Z80 PIO must also be set-up with software commands to select the direction of signal flow in the Z80 PIO.

SOFTWARE

PROGRAMMING EXAMPLE

Lets assume that you have a paper tape punch that you want to connect to the 820 through the Parallel Port. This punch has eight Data Bits And a strobe as its inputs from the 820. The output of the punch is a ready signal which will be low when ready to receive characters. The strobe will be software generated on bit 2 of the B side.

This sub-routine would have to be executed once to initialize the Z80 PIO.

```

GPACON      EQU      09          ;general purpose PIO A control
GPADAT      EQU      08          ;general purpose PIO A data
GPBCON      EQU      0BH        ;general purpose PIO B control
GPBDAT      EQU      0AH        ;general purpose PIO B data

INTPIO:     LD        C,GPACON   ;Port # to C register
            LD        B,3        ;Output 3 bytes
            LD        HL,GPPIO    ;Point HL & I to table
            OTIR       ;Output & Increment HL
            LD        C,GPBCON   ;Port # to C again
            LD        B,3        ;Output 3 bytes
            OTIR       ;Output & Increment HL
            LD        A,OFH      ;Strobe starts off high
            OUT        (GPBDAT),A ;Send to PIO B data
            RET              ;return to main program

GPIO:       ;PORT A
            DEFB       07        ;Disable interrupts
            DEFB       0CFH      ;Set port A to mode 3
            DEFB       00        ;Make all eight bits outputs
            ;PORT B
            DEFB       07        ;Disable interrupts
            DEFB       0CFH      ;Set port B to mode 3
            DEFB       0F0H      ;Bits 4-7 inputs, bits 0-3 outputs

```

This sub-routine will output the character in the C register to the paper tape punch.

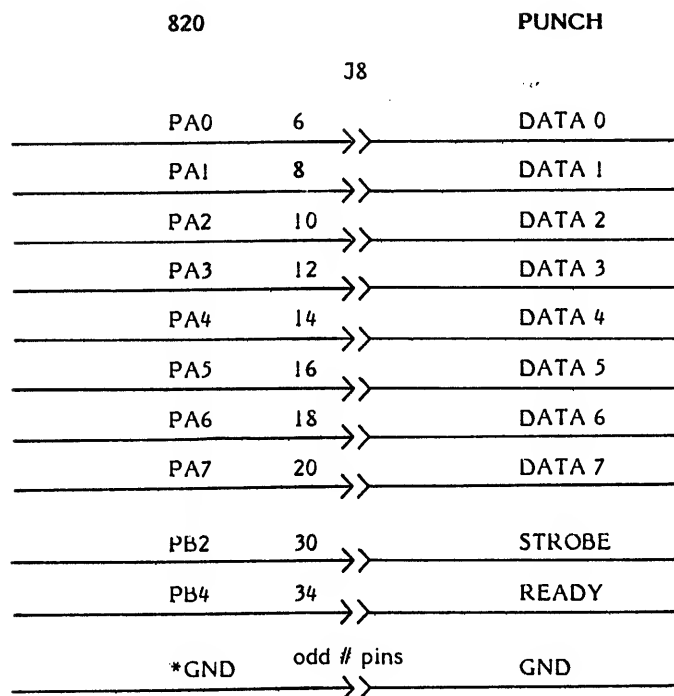
```

PUNOUT:     IN         A,(GPBDAT) ;read port B into A register
            AND        IOH        ;mask out all but ready
            JR         NZ,PUNOUT  ;repeat until Punch is ready
            LD         A,C        ;get character to A register
            OUT        (GPADAT),A ;send character to punch
            IN         A,(GPBDAT) ;read channel B to A register
            RES        2,A        ;make strobe line low
            OUT        (GPBDAT),A
            SET        2,A        ;make strobe line high
            OUT        (GPBDAT),A
            RET              ;back to calling routine

```

SOFTWARE

The paper tape punch would be connected to the Parallel Port as follows:



* On ETCH I CPU Board, pick up ground for J8 on odd pins of J11.

The following jumpers would be installed on J11:

PINS	FUNCTION
9-10	Select output for high nibble of Port A
17-18	Select output for low nibble of Port A
5-6	Select output for low nibble of Port B

TIMER

The Xerox 820 is equipped with a Z80 CTC (Counter Timer Circuit). The CTC has four independent channels that perform counting and timing functions. Channels 1, 2 and 3 are used by the 820's monitor.

Channel 0 is not used and can be configured to perform counting or timing functions for your program.

The following example is when the CTC might be used and some programming examples to help you understand its operation.

Lets assume that you are writing a program that among other things, samples an input signal that is connected to a temperature sensitive switch located on your manufacturing line. When this input goes to a logic 1 (+ 5 volts) you want the 820 to activate an alarm (also connected to the parallel port) by making an output signal a logic 1 (+ 5 volts), also you want to display a message on the 820's screen to inform the operator that there is a fire on the manufacturing line. Lets say that you have determined that this input needs to be looked at about 60 times each second.

A simple solution would be to use the Z80 CTC channel 0 and program it to give the 820 an interrupt every 16.69 milliseconds. Your interrupt service routine would look at the input and if it is a logic 1 (+ 5 volts) activate the alarm and display the message on the screen.

Lets assume that the input signal comes into the 820's General Purpose Port channel A on bit 7, and the alarm is connected to channel A on bit 0. You would have to install a jumper on J11 between pins 17 and 18 to select bits 0 - 3 as outputs.

Listed below is an example of how to:

- Initialize the CTC as a timer to generate an interrupt signal every 16.69 milliseconds.
- Write an interrupt service routine for the Z80 CTC.
- Disable the Z80 CTC's interrupt before exiting the program.

.Z80

CTCVEC	EQU	0FF10H	;CTC0 VECTOR LOCATION IN TABLE
GPACON	EQU	09	;GP PIO CHANNEL A CONTROL PORT
GPADAT	EQU	08	;GP PIO CHANNEL A DATA PORT
CTC0	EQU	18H	;CTC CHANNEL 0 PORT #
SENSOR	EQU	10000000B	;SENSOR BIT
ALARM	EQU	00000001B	;ALARM BIT
BDOS	EQU	5	;BDOS ENTRY POINT
WBOOT	EQU	0	;CP/M WARM ENTRY POINT
PRTSTG	EQU	9	;CP/M PRINT STRING FUNCTION
CLRSCN	EQU	1AH	;820 CLEAR SCREEN CHARACTER

```
;
; YOU MUST EXECUTE THE "INIT" ROUTINE BEFORE THE CTC-0 WILL START
; GENERATING INTERRUPTS. TYPICALLY THIS WOULD BE BEFORE THE MAIN
; BODY OF YOUR PROGRAM.
;
```

```
CALL INIT ;GO INITIALIZE CTC-0 & GP PIO
```

```
;*****
;
; MAIN PROGRAM
;
;*****
```

```
;
;-----THE MAIN PART OF YOUR PROGRAM WOULD GO HERE.
;
```

SOFTWARE

```

;
;-----IF YOUR PROGRAM NEEDS TO TERMINATE & GO BACK TO CP/M
;-----IT SHOULD DO SO BY JUMPING TO THE EXIT ROUTINE.
;-----THIS WILL DISABLE THE CTC-0 INTERRUPTS
;

```

```

JP      EXIT      ;EXIT TO CP/M

```

```

;
; INITIALIZATION SUBROUTINE - THIS ROUTINE WILL BE EXECUTED ONCE AT
; THE BEGINNING OF YOUR PROGRAM.
;
; 1 - STORE INTERRUPT SERVICE ROUTINE ADDRESS IN THE MODE 2 INTERRUPT
;     TABLE (FF10).
; 2 - SET UP THE PIO TO MONITOR THE SENSOR AND CONTROL THE ALARM
; 3 - SET UP THE CTC CHANNEL 0 TO GENERATE AN INTERRUPT EVERY 16.69
;     MILLISECONDS.
;

```

```

INIT:    LD      HL,INTROU      ;INTERRUPT ROUTINE'S ADDRESS
         LD      (CTCVEC),HL   ;SAVE IN INTERRUPT TABLE
         LD      C,GPAICON     ;GP PIO CHANNEL A CONTROL PORT
         LD      B,4           ;OUTPUT FOUR BYTES
         LD      HL,PIOTBL     ;START OF PIO TABLE
         OTIR                ;SEND TABLE TO PIO
         XOR     A             ;A REGISTER = 00
         OUT     (GPADAT),A    ;MAKE ALARM OUTPUT = 00
         LD      C,CTC0        ;CTC CHANNEL 0 PORT # TO C
         LD      B,2           ;OUTPUT TWO BYTES
         LD      HL,CTCTBL     ;START OF CTC TABLE
         OTIR                ;SEND TO CTC
         RET                  ;BACK TO CALLER

```

```

;PIO INITIALIZATION TABLE

```

```

PIOTBL:  DEFB    00           ;DISABLE INTERRUPTS
         DEFB    0CFH        ;SET TO MODE 3
         DEFB    0F0H        ;BIT 4-7 = INPUT'S
         DEFB    07          ;INTERRUPT SWITCH

```

```

;CTC INITIALIZATION TABLE

```

```

CTCTBL:  DEFB    10100111B   ;CTC 0 SET TO TIMER MODE
         DEFB    163         ;CTC 0 PERIOD 163*256*400 NSEC.

```

```

;
; INTERRUPT SERVICE ROUTINE -
; THIS ROUTINE WILL BE EXECUTED ONCE EVERY 16.69 MILLISECONDS.
; IT WILL DO THE FOLLOWING:
; 1 - MONITOR SENSOR INPUT
; 2 - WHEN INPUT IS HIGH ACTIVATE ALARM AND DISPLAY MSG ON SCREEN.
;

```

```

INTROU:  PUSH    HL           ;SAVE REGISTERS
         PUSH    BC
         PUSH    DE
         PUSH    AF
         IN      A,(GPADAT)   ;READ GP PIO CHANNEL A DATA
         AND     SENSOR       ;MASK ALL BUT SENSOR INPUT
         JR      Z,NOTHOT     ;IF RESULT = 00 - NO FIRE
         LD      A,ALARM      ;ELSE SOUND ALARM
         OUT     (GPADAT),A   ;ACTIVATE ALARM
         LD      C,PRTSTG     ;PRINT STRING FUNCTION
         LD      DE,MESG1     ;POINT TO MESSAGE
         CALL    BDOS
         JR      OUT1         ;EXIT INTERRUPT ROUTINE

```

SOFTWARE

```

MSG1:  DEFB  CLRSCN
       DEFM  ' *** FIRE ON MANUFACTURING LINE *** '
       DEFM  '$'

NOTHOT: XOR   A           ;CLEAR A REGISTER
       OUT   (GPADAT),A   ;TURN ALARM OFF

OUT1:   POP   AF           ;RESTORE REGISTERS
       POP   DE
       POP   BC
       POP   HL
       EI           ;ENABLE INTERRUPTS
       RETI          ;RETURN FROM INTERRUPT

```

```

;
; ROUTINE THAT USER'S PROGRAM SHOULD JUMP TO WHEN IT IS READY
; TO EXIT BACK TO CP/M. THIS ROUTINE DISABLES THE CTC0 INTERRUPT
; AND DOES A CP/M WARM BOOT.
;

```

```

EXIT:   LD     A,01         ;PREPARE TO DISABLE CTC
       DI           ;DISABLE INTERRUPTS
       OUT   (CTC0),A      ;SEND TO CTC-0
       EI           ;INTERRUPTS OK NOW
       JP    0            ;BACK TO CP/M

       END

```

REAL TIME CLOCK

The following program is the Z80 assembly listing for a Real Time Clock. This program can be entered assembled and run on your 820 without making any hardware modifications or additions.

Features: 12 or 24 hour format

Time can be displayed on the screen if desired

Memory locations that store the time can be accessed from other programs to read the current time.

NOTE: This clock increments the seconds every .999936 micro-seconds. This along with tolerances in the system master oscillator will effect the accuracy of the clock. Typically over a 24 hour period it may gain or loose as much as 20 seconds.

You will need the following to create and assemble the program:

Text Editor (such as, Xerox Word Processing)

M80.COM (Z80 assembler on CP/M disk)

L80.COM (Linker on CP/M disk)

First you will need to enter this program with a text editor and name the file CLOCK.MAC. If you are using the Xerox Word Processing, choose the E command - edit a program from the directory menu. When the program has been entered, run the assembler by entering : M80 CLOCK,CLOCK=CLOCK. When the assembly process is complete you should get a message that there were no fatal errors. If you do not get this message, check your typing for errors. Next you will link your file by entering : L80 CLOCK,CLOCK/N/E. This will generate a file named CLOCK.COM on your disk.

You can now execute the clock program by entering: CLOCK (RET) This brings up a screen of instructions on what to enter to activate and set the clock. For example, if you entered CLOCK SD093000 the clock would be set for standard time, display the time on the screen and set the time for 9:30:00. If you enter clock after the program has been loaded, it will come back and tell you what memory locations the hours, minutes and seconds are stored at.

```
.Z80
BDOS      EQU    5           ;BDOS ENTRY POINT
PRTSTG    EQU    9           ;CP/M PRINT STRING FUNCTION
CLRSCN    EQU    1AH        ;CLEAR SCREEN CODE
CR        EQU    0DH        ;CARRIAGE RETURN CODE
LF        EQU    0AH        ;LINE FEED CODE
CTC3      EQU    0FF16H     ;CTC CHANNEL 3 INTERRUPT VECTOR
BASE1     EQU    0FF75H     ;BASE VARIABLE (1.0 ROM)
BASE2     EQU    0FF78H     ;BASE VARIABLE (2.0 ROM)
CLKORG    EQU    0FE00H     ;ORIGIN FOR CLOCK ROUTINE
HOURS     EQU    0FF5CH     ;HOURS VARIABLE
MINUTE    EQU    HOURS + 1   ;MINUTES VARIABLE LOCATION
SECNDS    EQU    MINUTE + 1  ;SECONDS VARIABLE LOCATION

;
;FIRST CHECK FOR CLOCK MODULE ALREADY LOADED, IF IT IS DISPLAY
;MESSAGE AND GO BACK TO CP/M
;

BEGIN:    LD      A, (0FE00H)
          CP      0FFH       ;CHECK FOR CLOCK ALREADY LOADED
          JR      Z,PROCED   ;PROCEED IF NOT
          LD      DE,RESET   ;POINT TO RESET MESSAGE
          LD      C,PRTSTG   ;PRINT STRING FUNCTION TO C
          CALL    BDOS       ;CALL BDOS
          RST     0          ;BACK TO CP/M
```

SOFTWARE

```

;
;CHECK LENGTH OF COMMMAND LINE (MUST BE 9 CHARACTERS) IF NOT GIVE
;USER INSTRUCTIONS ON WHAT MUST BE ON COMMAND LINE
;

```

```

PROCED:  LD      A,(80H)      ;GET COMMAND LINE LENGTH
        CP      9            ;CHECK FOR 9 CHARACTERS
        JP      Z,PARMOK     ;IF COUNT = 9 THEN GO AHEAD
        LD      DE,INSTR     ;ELSE PRINT INSTRUCTIONS
        LD      C,PRTSTG     ;PRINT STRING FUNCTION TO C
        CALL    BDOS         ;GO PRINT THROUGH CP/M
        RST     0            ;GO BACK TO CP/M

```

```

;
; IF CLOCK IS NOT LOADED AND COMMAND LINE PARAMETER COUNT IS OK
; MOVE IMAGE OF CLOCK ROUTINE TO HIGH MEMORY
;

```

```

PARMOK:  LD      HL,START     ;SOURCE ADDRESS FOR MOVE
        LD      DE,CLKORG     ;DESTINATION ADDRESS FOR MOVE
        LD      BC,LENGTH     ;NUMBER OF BYTES TO MOVE
        LDIR                     ;Z-80 BLOCK MOVE
        DI                     ;DISABLE INTERRUPTS
        LD      HL,(CTC3)     ;ADDRESS OF 1 SEC. INTERRUPT ROUTINE
        LD      DE,12         ;OFFSET INTO ROUTINE
        ADD     HL,DE         ;COMPUTE ADDRESS
        LD      E,(HL)        ;GET LOW BYTE OF CALL TO E
        INC     HL            ;BUMP POINTER
        LD      D,(HL)        ;GET HIGH BYTE OF CALL TO D
        DEC     HL            ;ROLL HL BACK
        LD      BC,CLOCK      ;GET ADDRESS OF CLOCK ROUTINE
        LD      (HL),C        ;RE-ROUTE INTERRUPT TO CLOCK ROUTINE
        INC     HL
        LD      (HL),B
        LD      HL,GETOUT+1   ;POINT TO CLOCK EXIT
        LD      (HL),E        ;SAVE ORIGINAL LOW BYTE
        INC     HL
        LD      (HL),D        ;SAVE ORIGINAL HIGH BYTE
        LD      A,(0F001H)    ;GET BYTE FROM MONITOR
        CP      45H          ;CHECK FOR 2.0 ROM
        JR      NZ,ROM1       ;SKIP IF NOT
        LD      HL,BASE2      ;NEW BASE ADDRESS
        LD      (CLOCK+7),HL  ;SAVE NEW VALUE
ROM1:    LD      A,(82H)       ;GET STD/MILITARY OPTION
        CP      'M'          ;CHECK FOR M
        JR      NZ,BASEOK     ;DEFAULT STD TIME SKIP OVER
        LD      A,25D         ;NEW VALUE
        LD      (BASE+1),A    ;SAVE NEW VALUE
BASEOK:  LD      A,(83H)       ;GET DISPLAY OPTION
        CP      'N'          ;CHECK FOR NO DISPLAY
        JR      NZ,DISOK      ;DEFAULT ON SKIP AROUND
        LD      A,0C3H        ;GET JUMP INSTRUCTION
        LD      (CLOCK+6),A   ;SAVE IN PLACE OF CALL
        LD      HL,(GETOUT+1)
        LD      (CLOCK+7),HL
DISOK:   LD      HL,(84H)     ;GET HOURS VALUE
        CALL    CONV          ;GO CONVERT TO BINARY
        LD      (HOURS),A     ;SAVE IN HOURS VARIABLE
        LD      HL,(86H)     ;GET MINUTES VALUE
        CALL    CONV          ;GO CONVERT TO BINARY
        LD      (MINUTE),A    ;SAVE IN MINUTES VARIABLE
        LD      HL,(88H)     ;GET SECONDS VALUE
        CALL    CONV          ;GO CONVERT TO BINARY
        LD      (SECNDS),A    ;SAVE IN SECONDS VARIABLE
        EI
        LD      A,1AH        ;CLEAR SCREEN CODE
        CALL    0F00FH        ;GO THROUGH MONITOR
        RET

```

SOFTWARE

```

;
; CONVERT ASCII VALUE IN H&L TO BINARY VALUE & RETURN IN A REGISTER.
; UNITS IN H -- TENS IN L
;

```

```

\ CONV:      LD      A,H          ;MOVE TO A
             SUB     30H         ;REMOVE ASCII OFFSET
             LD      H,A         ;PUT BACK IN H
             LD      A,L         ;MOVE L TO A
             SUB     30H         ;REMOVE ASCII OFFSET
             LD      L,A         ;PUT BACK IN L
             ADD     A,A         ;DOUBLE A
             ADD     A,A         ;DOUBLE AGAIN
             ADD     A,L         ;ADD ONE IN
             ADD     A,A         ;A = A * 10
             ADD     A,H         ;ADD IN UNITS VALUE
             RET              ;ALL DONE

```

```

;
; MAIN CLOCK ROUTINE - THIS CODE IS MOVED INTO HIGH MEMORY AND EXECUTED
; EVERYTIME A ONE SECOND INTERRUPT OCCURS
;

```

```

START:
CLOCK:      .PHASE CLKORG
            LD      HL,SECNDS    ;POINT HL TO SECONDS VARIABLE
            CALL    INCTIM       ;INCREMENT TIME IN BINARY
            LD      A,(BASE1)    ;GET LINE# OF BOTTOM LINE ON SCREEN
            INC     A            ;ADD 1 TO WRAP AROUND TO TOP LINE
            CP      24
            JR      C,CLOCK2     ;WATCH FOR MODULO 24 THING
            XOR     A
CLOCK2:     SRL     A            ;TRANSFORM LINE# INTO 16 BIT ADDRESS
            LD      L,70*2       ;WITH COL# COMPONENT=70
            RR      L
            LD      DE,3000H
            OR      D
            LD      H,A
            IN      A,(1CH)
            SET     7,A
            OUT     (1CH),A      ;ENABLE CRT RAM BANK
            LD      DE,HOURS     ;POINT DE TO CLOCK HOURS
            LD      (HL),' '
            INC     HL
            CALL    PUTDEC       ;CALL PUTDEC TO DISPLAY HOURS
            LD      (HL),' '
            INC     HL
            CALL    PUTDEC       ;CALL PUTDEC TO DISPLAY MINUTES
            LD      (HL),' '
            INC     HL
            CALL    PUTDEC       ;CALL PUTDEC TO DISPLAY SECONDS
            LD      (HL),' '
            IN      A,(1CH)
            RES     7,A
            OUT     (1CH),A      ;DISABLE CRT ROM BANK
GETOUT:     JP      0

```

```

;
; SUBROUTINE TO PUT DECIMAL CONTENTS OF CLOCK VARIABLE LOCATIONS ON THE
; SCREEN. ENTER WITH THE DE REGISTER POINTING TO THE DESIRED VARIABLE
;

```

SOFTWARE


```

PUTDEC:  LD  A,(DE)
         INC DE
         LD  C,0
PUTD1:   SUB 10
         JR  C,PUTD2
         INC C
         JR  PUTD1
PUTD2:   ADD A,10
         PUSH AF
         LD  A,C
         CALL PUTDIG      ;DISPLAY 10'S DIGIT OF TIME
         POP AF
PUTDIG:  OR  '0'          ;MAKE MSB OF ACC INTO ASCII
         LD  (HL),A
         INC HL           ;STORE CHARACTER AND BUMP POINTER
         RET

```

```

;
; INCREMENT TIME IN SECONDS VARIABLE BY ONE, CHECK FOR:
; SECONDS = 59, MINUTES = 59, AND HOURS = 12.
;

```

```

INCTIM:  INC  (HL)
         LD  A,(HL)
         CP  60
         RET C           ;BUMP CLOCK SECONDS AND CHECK FOR
                        ; ROLL-OVER AT END OF MINUTE
         LD  (HL),0
         DEC HL          ;EXIT IF NO CARRY TO MINUTES
                        ;ELSE RESET SECONDS TO ZERO
         INC (HL)
         LD  A,(HL)
         CP  60
         RET C           ; AND POINT NEXT TO MINUTES
                        ;BUMP CLOCK MINUTES AND CHECK FOR
                        ; ROLL-OVER AT END OF HOUR
         LD  (HL),0
         DEC HL          ;EXIT IF NO CARRY INTO HOURS
                        ;ELSE RESET MINUTES TO ZERO
         INC (HL)
                        ; AND POINT NEXT TO HOURS
         LD  A,(HL)
         CP  13
         RET C           ;BUMP CLOCK HOURS AND CHECK FOR
                        ; ROLL-OVER AFTER 24 HOURS
         LD  (HL),1
         RET             ;EXIT IF NO ROLL-OVER
                        ;ELSE RESET HOURS TO 1 AND
                        ;START OVER

LENGTH  EQU  $-CLOCK    ;CALCULATE LENGTH OF CODE
        .DEPHASE

```

```

;
; MESSAGES
;

```

```

INSTR:   DEFB CLRSCN,LF
         DEFM '                CLOCK UTILITY INSTRUCTIONS'
         DEFM ' VER 1.0'
         DEFB CR,LF,LF
         DEFM 'THE COMMAND LINE TO SET & RUN THE CLOCK MUST BE AS '
         DEFM 'FOLLOWS:'
         DEFB CR,LF,LF
         DEFM 'A CLOCK ABHHMMSS'
         DEFB CR,LF,LF
         DEFM '  A = S FOR STANDARD TIME'
         DEFB CR,LF
         DEFM '          M FOR MILITARY TIME'
         DEFB CR,LF,LF
         DEFM '  B = D TO DISPLAY TIME ON SCREEN'
         DEFB CR,LF
         DEFM '          N NO DISPLAY ON SCREEN'
         DEFB CR,LF,LF

```

SOFTWARE

```

DEFM ' HH = HOUR'
DEFB CR,LF,LF
DEFM ' MM = MINUTE'
DEFB CR,LF,LF
DEFM ' SS = SECOND'
DEFB CR,LF,LF,LF
DEFM '$'

RESET: DEFB CLRSCN,LF,LF
DEFM 'THE CLOCK MODULE IS ALREADY LOADED, PRESS RESET'
DEFM 'IF YOU WANT TO RELOAD IT.'
DEFB CR,LF,LF,LF,LF,LF,LF,LF,LF,LF
DEFB ' CLOCK VARIABLE MEMORY LOCATIONS'
DEFB CR,LF,LF,LF
DEFM ' DECIMAL HEX VARIABLE'
DEFB CR,LF,LF
DEFM ' 65372 FF5C HOURS'
DEFB CR,LF
DEFM ' 65373 FF5D MINUTES'
DEFB CR,LF
DEFM ' 65374 FF5E SECONDS'
DEFB CR,LF,LF,LF
DEFM '$'

END BEGIN

```

CBIOS MODIFICATION PROCEDURE

Procedure to generate a new system (CP/M) disk after making modifications to your CBIOS (level 2.0 CP/M disks and later). NOTE - underscored text indicates entered by you, (RET) means press the return key.

You should have the following files on your disk:

M80.COM	-	Macro - 80 Assembler
L80.COM	-	Link - 80 Linker
DDT.COM	-	Dynamic Debugging Tool
SYSGEN.COM	-	System Generation Utility
CBIOS.MAC	-	Source File for CBIOS

Assemble your source file (CBIOS.MAC) by entering the following:

A > M80 CBIOS,CBIOS=CBIOS (RET)

When the assembly process is complete, you should be prompted with the message NO FATAL ERRORS. If the assembler detects any errors, you should correct them and re-assemble your source file before proceeding.

If you have made additions to the CBIOS you should type out the list file to determine if it has exceeded the amount of space remaining on the disk. The file can be displayed by entering:

A > TYPE CBIOS.LST (Ret)

On an 8.0" system, you have 896 Bytes available for the CBIOS, a 5.25" system has 1152 bytes available. Currently the end of the code is three lines after the label XEROXID:. The DEFB '\$' three lines after the XEROXID label should not have an address higher than 895 (37F hex) on an 8" system or 1151 (47F hex) on a 5.25" system.

```
44C  20 20 20 20  XEROXID:  DEFM  '      '
      20 20 20 20
      20
455  OD OA                DEFB  CR,LF
454  24                  DEFB  '$'
```

Check this address -

It should be less than 480 (hex) for a 5.25" system and less than 380 (hex) for an 8" system.

If the above example were an actual listing, it would work OK on a 5.25" system but not on an 8".

Now use L80 to create CBIOS.HEX by entering the following:

A > L80 CBIOS/P:EA00,CBIOS/N/X/E (Ret)

L80 will ask you for a Y or N Input, you should respond with:

N (Ret)

You should now have CBIOS.HEX in your directory.

Use sysgen to get an image of your present operating system in the directory by entering:

A > SYSGEN
Sysgen Ver 2.0
Source Drive Name (or Return to Skip) A
Source on A, then type Return (Ret)
Destination Drive (or Return to Reboot) (Ret)

A > SAVE 34 CPM.COM

SOFTWARE

Use DDT to "overlay" your new CBIOS over the previous one by entering the following:

```
A> DDT CPM.COM (Ret)
  DDT  VER  2.2
  Next  PC
  2300  100
-ICBIOS.HEX (Ret)
-R 3580 (Ret)
  Next  PC
  XXXX  0000
-GO (Ret)
```

Now, execute sysgen again to record your newly modified system on a disk.

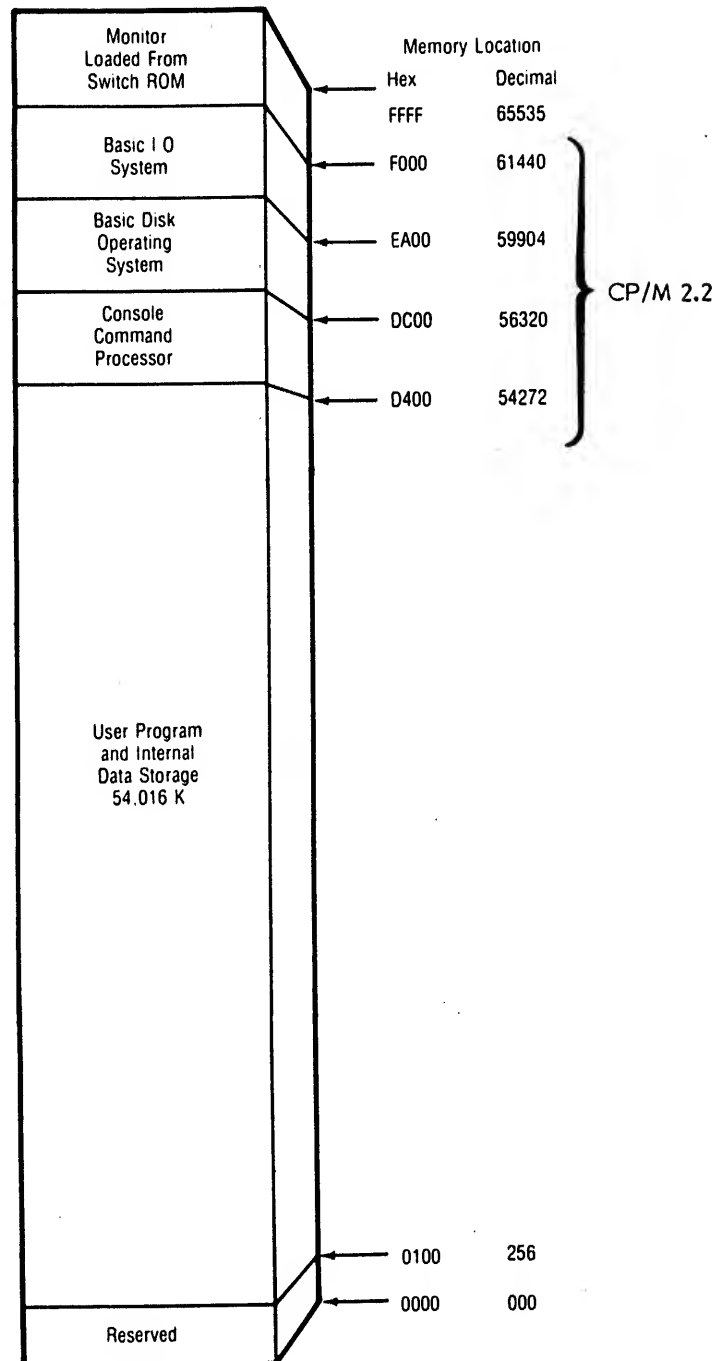
```
A> SYSGEN (Ret)
```

```
Sysgen Ver 2.0
Source Drive Name (or Return to Skip)  (Ret)
Destination Drive (or Return to Reboot) A
Destination on A, then Type Return      (Ret)
Destination Drive (or Return to Reboot) (Ret)
```

You must press the reset button in the rear of the 820 and "COLD BOOT" from your newly modified disk.

MEMORY ORGANIZATION

MEMORY MAP FOR CP/M SYSTEM



CP/M is a registered trade mark of Digital Research, Inc.

SOFTWARE

SOFTWARE

3 - 32

PROGRAM LISTINGS

MONITOR ROM VERSION 1.0 (U64 + U63)

```

0001 ;*****
0002 ;*
0003 ;*          XEROX 820    MONITOR ROM          *
0004 ;*
0005 ;*          VERSION 1.0
0006 ;*
0007 ;*****
0008 ;
0009 ;
0010 PSECT ABS
EFF0 0011 ROM EQU OEFF0H ;START OF 4K ROM-TRANSFER CODE
FF00 0012 RAM EQU OFF00H ;START OF 256 BYTE RAM
3000 0013 CRTMEM EQU 3000H ;BASE OF 4K CRT MEMORY
0014 ;
0015 ;
EFF0 0016 ORG ROM
0017 ;
0018 ;
0019 ; COPY ROM CODE TO HIGH MEMORY
0020 ; ON POWER-UP
0021 ;
EFF0 F3 0022 DI ;KEEP OTHERS AWAY
EFF1 211000 0023 LD HL,0010H ;SET START ADDRESS
EFF4 1100F0 0024 LD DE,0F000H ;SET DESTINATION ADDRESS
EFF7 010010 0025 LD BC,1000H ;SET LENGTH OF MOVE
EFAA EDB0 0026 LDIR ;MOVE IT ALL
EFFC C300F0 0027 JP OF000H ;JUMP TO THE ROM CODE IN HI MEM
EFFF 00 0028 NOP ;JUST TO LINE UP BOUNDS
0029 ;
0030 ;
0031 INCLUDE INIT.ASM
0032 ;*****
0033 ;*
0034 ;* COLD START INITIALIZATION ROUTINE FOR
0035 ;* CONFIGURING THE SYSTEM AFTER A POWER-ON
0036 ;* OR PUSHBUTTON RESET.
0037 ;*
0038 ;*****
0039 ;
0040 ;
0041 ; -- MONITOR ENTRY POINT TABLE --
0042 ;
F000 C32AF0 0043 COLD: JP INIT ;MONITOR COLD ENTRY POINT
F003 C3EDF0 0044 WARM: JP PROMPT ;MONITOR WARM ENTRY POINT
F006 C398F5 0045 CONST: JP KBDST ;CONSOLE STATUS VECTOR
F009 C3A0F5 0046 CONIN: JP KBDIN ;CONSOLE INPUT VECTOR
F00C C34BF6 0047 CONOUT: JP CRTOUT ;CONSOLE OUTPUT VECTOR
F00F C34BF6 0048 JP CRTOUT ;CRT OUTPUT VECTOR
F012 C32EF6 0049 JP SIOST ;SIO CHANEL B STATUS VECTOR
F015 C336F6 0050 JP SIOIN ;SIO CHANEL B INPUT VECTOR
F018 C340F6 0051 JP SIOOUT ;SIO CHANEL B OUTPUT VECTOR
F01B C3DCF7 0052 JP SELECT ;DISK DRIVE SELECT
F01E C312F8 0053 JP HOME ;HOME R/W HEAD
F021 C324F8 0054 JP SEEK ;SEEK TO TRACK
F024 C35FF8 0055 JP READ ;READ SECTOR
F027 C351F8 0056 JP WRITE ;WRITE SECTOR
0057 ;
0058 ;
0059 ;
0060 ; DO A SHORT POST-RESET DELAY BY FILLING THE
0061 ; 256 BYTE SCRATCH MEMORY WITH ZEROS
0062 ;
F02A F3 0063 INIT: DI

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

F02B 2100FF 0064 LD HL, RAM ;POINT TO START OF MONITOR RAM
F02E 3600 0065 INIT1: LD (HL),0 ;FILL 256 BYTE SPACE WITH ZEROS
F030 F9 0066 LD SP, HL ;DO SOMETHING USEFUL TO ADD DELAY
F031 2C 0067 INC L
F032 20FA 0068 JR NZ, INIT1-$ ;LOOP TAKES ABOUT 4 MILLISECONDS
0069 ;
0070 ; STORE ANY NON-ZERO VALUES FOR VARIABLES IN MEMORY
0071 ;
F034 21A1F0 0072 LD HL, INTAB ;POINT TO DEFAULT VARIABLE TABLE
F037 0600 0073 INIT2: LD B, 0
F039 4E 0074 LD C, (HL) ;BC=DATA BLOCK BYTECOUNT
F03A 23 0075 INC HL
F03B 5E 0076 LD E, (HL) ;DE=DESTINATION FOR DATA
F03C 23 0077 INC HL
F03D 56 0078 LD D, (HL)
F03E 23 0079 INC HL
F03F EDB0 0080 LDIR ;COPY DATA @ HL TO VARIABLES @ DE
F041 CB7E 0081 BIT 7, (HL)
F043 28F2 0082 JR Z, INIT2-$ ;LOOP AGAIN IF NOT AT END OF TBL
0083 ;
0084 ; INITIALIZE THE PROGRAMMABLE I/O DEVICES
0085 ;
F045 23 0086 INC HL ;POINT TO I/O INIT DATA TABLE
F046 46 0087 INIT3: LD B, (HL) ;B=INIT LOOP BYTECOUNT
F047 23 0088 INC HL
F048 4E 0089 LD C, (HL) ;C=DEVICE CONTROL PORT#
F049 23 0090 INC HL
F04A EDB3 0091 OTIR ;SEND DATA @ HL TO PORT @ C
F04C CB7E 0092 BIT 7, (HL) ;TEST FOR TABLE END MARKER
F04E 28F6 0093 JR Z, INIT3-$ ;LOOP AGAIN IF NOT AT END
0094 ;
0095 ; INITIALIZE THE Z-80 FOR INTERRUPT MODE #2
0096 ;
F050 3EFF 0097 LD A, VECTAB.SHR.8
F052 ED47 0098 LD I, A ;LOAD I REG WITH MSB OF VECTOR TBL
F054 ED5E 0099 IM 2 ; AND SELECT INTERRUPT MODE 2
0100 ;
0101 ; PRINT SIGNON MESSAGE
0102 ;
F056 FB 0103 SIGNON: EI
F057 CDE4F3 0104 CALL PNEXT
F05A 1A 0105 DEFB 'Z'-64
F05B 2E2E2E58 0106 DEFM '...XEROX 820 VER. 1.0...'
45524F58
20383230
20205645
522E2031
2E302E2E
2E
F074 0D0A 0107 DEFB CR, LF
F076 20202041 0108 DEFM ' A - BOOT SYSTEM'
202D2042
4F4F5420
53595354
454D
F088 0D0A 0109 DEFB CR, LF
F08A 20202054 0110 DEFM ' T - TYPEWRITER'
202D2054
59504557
52495445
52
F09B 0D0A 0111 DEFB CR, LF
F09D 04 0112 DEFB EOT
F09E C303F0 0113 JP WARM ;GO ENTER MONITOR
0114 ;
0115 ;
0116 ;

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

FOA1      0117 ;
          0118 INTAB EQU $ ;INITIALIZATION DATA TABLES
          0119 ;
          0120 ; INITIALIZE THE Z-80 'I' REGISTER INTERRUPT VECTOR TABLE
          0121 ;
FOA1 02   0122 DEFB 2
FOA2 1AFF 0123 DEFW SYSVEC+2
FOA4 DEF5 0124 DEFW KEYSRV ;PARALLEL KEYBD INTERRUPT VECTOR
          0125 ;
FOA6 02   0126 DEFB 2
FOA7 12FF 0127 DEFW CTCVEC+2
FOA9 15F6 0128 DEFW MILLI ;ONE MILLISECOND INTERRUPT TIMER
          0129 ;
FOAB 02   0130 DEFB 2
FOAC 16FF 0131 DEFW CTCVEC+6
FOAE FCF5 0132 DEFW TIMER ;ONE SECOND TIMER INTERPT VECTOR
          0133 ;
          0134 ; INITIALIZE DISK I/O DRIVER VARIABLES
          0135 ;
FOB0 08   0136 DEFB 8
FOB1 5FFF 0137 DEFW UNIT
FOB3 FF   0138 DEFB 255 ;FLAG ALL DRIVES AS DE-SELECTED
FOB4 FFFFFFFF 0139 DEFB 255,255,255,255 ;CLEAR HEAD POSITION TABLE
FOB8 03   0140 DEFB 00000011B ;SELECT SLOWEST SEEK SPEED
FOB9 80   0141 DEFB 128 ;SELECT 128 BYTE SECTOR LENGTH
FOBA 0F   0142 DEFB 15 ;SET MOTOR TURN-OFF TIMER
          0143 ;
          0144 ; INITIALIZE THE CRT DISPLAY CURSOR
          0145 ;
FOBB 01   0146 DEFB 1
FOBC 74FF 0147 DEFW CSRCHR
FOBE 02   0148 DEFB 02 ;USE NON-BLINKING BOX
          0149 ;
          0150 ; SET FREE MEMORY POINTER
          0151 ;
FOBF 02   0152 DEFB 2
FOC0 77FF 0153 DEFW FREPTR
FOC2 69F9 0154 DEFW ROMEND ;POINT TO FIRST LOCATN AFTER MONITR
          0155 ;
          0156 ;
FOC4 FF   0157 DEFB -1 ;END OF VARIABLE INIT TABLE
          0158 ;
          0159 ;
          0160 ;
0000      0161 BAUDA EQU 00H ;CHANEL A BAUD RATE GENETATOR
0004      0162 SIO EQU 04H ;DUAL SERIAL I/O
0008      0163 GENPIO EQU 08H ;GENERAL PURPOSE PARALLEL I/O
000C      0164 BAUDB EQU 0CH ;CHANEL B BAUD RATE GENERATOR
0010      0165 WD1771 EQU 10H ;WESTERN DIGITAL DISK CONTROLLER
0014      0166 SCROLL EQU 14H ;CRT SCROLL MEMORY SCROLL REG
0018      0167 CTC EQU 18H ;QUAD COUNTER/TIMER CIRCUIT
001C      0168 SYSPIO EQU 1CH ;SYSTEM PARALLEL I/O
          0169 ;
          0170 ; INITIALIZE SYSTEM PIO FOR USE AS BANK-SWITCH,
          0171 ; DISK DRIVE SELECT AND PARALLEL KEYBOARD INPUT
          0172 ;
001C      0173 BITDAT EQU SYSPIO+0
001D      0174 BITCTL EQU SYSPIO+1
001E      0175 KBDDAT EQU SYSPIO+2
001F      0176 KBDCTL EQU SYSPIO+3
          0177 ;
FOC5 031D 0178 DEFB 3,BITCTL
FOC7 CF   0179 DEFB 11001111B ;PUT SYSTEM PIO IN BIT MODE
FOC8 18   0180 DEFB 00011000B ;MAKE BITS 4 AND 3 BE INPUTS
FOC9 40   0181 DEFB 01000000B ;DISABLE INTERRUPTS
          0182 ;
FOCA 011C 0183 DEFB 1,BITDAT

```

```

FOCC 00      0184      DEFB      00000000B      ;DE-SELECT ROMS, ENABLE DRIVE 0
              0185 ;
FOCD 031F    0186      DEFB      3,KBDCTL
FOCF 4F      0187      DEFB      01001111B      ;PUT KEYBOARD PORT IN INPUT MODE
FODO 1A      0188      DEFB      SYSVEC+2      ;LOAD KEYBOARD INTERRUPT VECTOR
FOD1 83      0189      DEFB      10000011B      ;ENABLE INTERRUPTS
              0190 ;
              0191 ;
              0192 ;      INITIALIZE CHANELS 2 AND 3 OF THE CTC
              0193 ;      TO GENERATE ONE SECOND INTERRUPTS FROM CTC3
              0194 ;
0018      0195 CTC0 EQU      CTC+0      ;CTC CHANEL 0 PORT#
0019      0196 CTC1 EQU      CTC+1      ;CTC CHANEL 1
001A      0197 CTC2 EQU      CTC+2      ;CTC CHANEL 2
001B      0198 CTC3 EQU      CTC+3      ;CTC CHANEL 3
              0199
FOD2 0118    0200      DEFB      1,CTC0
FOD4 10      0201      DEFB      CTCVEC      ;BASE INTERRUPT VECTOR FOR CTC
              0202 ;
FOD5 021A    0203      DEFB      2,CTC2
FOD7 27      0204      DEFB      00100111B      ;PUT CTC2 IN TIMER MODE
FOD8 69      0205      DEFB      105      ;CTC2 PERIOD=105*256*400 NANOSCNDNS
              0206 ;
FOD9 021B    0207      DEFB      2,CTC3
FODB C7      0208      DEFB      11000111B      ;PUT CTC3 IN COUNTER MODE
FODC 5D      0209      DEFB      93      ;CTC3 PERIOD=999936 MICROSECONDS
              0210 ;
              0211 ;
              0212 ;      INITIALIZE SIO CHANEL B FOR ASYNCHRONOUS SERIAL
              0213 ;      INTERFACE TO PRINTER OR TERMINAL
              0214 ;
0004      0215 SIODPA EQU      SIO+0      ;SIO DATA PORT A
0005      0216 SIODPB EQU      SIO+1      ;SIO DATA PORT B
0006      0217 SIOCPA EQU      SIO+2      ;SIO CONTROL/STATUS PORT A
0007      0218 SIOCPB EQU      SIO+3      ;SIO CONTROL/STATUS PORT B
              0219
FODD 0A07    0220      DEFB      10,SIOCPB
FODF 04      0221      DEFB      4      ;SELECT REGISTER #4
FOE0 45      0222      DEFB      01000101B      ;16X CLOCK, 1 STOP BIT
FOE1 01      0223      DEFB      1      ;SELECT REGISTER #1
FOE2 04      0224      DEFB      00000100B      ;STATUS AFFECTS VECTOR
FOE3 03      0225      DEFB      3      ;SELECT REGISTER #3
FOE4 41      0226      DEFB      01000001B      ;7 BITS/RX CHARACTER
FOE5 05      0227      DEFB      5      ;SELECT REGISTER #5
FOE6 2A      0228      DEFB      00101010B      ;7 BITS/TX CHARACTER
FOE7 02      0229      DEFB      2      ;SELECT REGISTER #2
FOE8 00      0230      DEFB      SIOVEC      ;BASE SIO INTERRUPT VECTOR
              0231
FOE9 010C    0232      DEFB      1,BAUDB
FOEB 05      0233      DEFB      0101B      ;DEFAULT BAUD RATE=300
              0234
FOEC FF      0235      DEFB      -1      ;END-OF-TABLE
              0236 ;
              0237 ;
              0238 ;
              0239 ;
              0240      INCLUDE MONITOR.ASM
0241 ;*****
0242 ;*
0243 ;*      BASIC HEX MONITOR FOR Z-80 PROCESSORS
0244 ;*
0245 ;*****
0246 ;
0247 ;
0248 ;
0249 ;
FOED CDE4F3  0250      PROMPT: CALL      PNEXT

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

FOF0	0D0A	0251	DEFB	CR,LF	
FOF2	2A20	0252	DEFM	'* '	
FOF4	04	0253	DEFB	EOT	
FOF5	2184FF	0254	LD	HL,LINBUF	
FOF8	0E20	0255	LD	C,32	
FOFA	CD31F3	0256	CALL	GETLIN	;INPUT A BUFERED CONSOLE LINE
FOFD	3835	0257	JR	C,WHAT-\$;PRINT 'WHAT ?' IF INPUT ERROR
		0258			
FOFF	AF	0259	XOR	A	
F100	3281FF	0260	LD	(ESCFLG),A	
F103	CDF4F3	0261	CALL	CRLFS	
F106	3A84FF	0262	LD	A,(LINBUF)	;GET FIRST CHARACTER IN LINE
F109	FE0D	0263	CP	CR	
F10B	28E0	0264	JR	Z,PROMPT-\$;JUMP IF A NULL LINE
F10D	2144F1	0265	LD	HL,CMDTAB	;SEARCH FOR A MATCHING CHARACTER
F110	010D00	0266	LD	BC,CMSIZ/3	; IN COMMAND SEARCH TABLE
F113	CD56F3	0267	CALL	SEARCH	
F116	201C	0268	JR	NZ,WHAT-\$;TRY AGAIN IF SEARCH FAILS
F118	C5	0269	PUSH	BC	
F119	FD2185FF	0270	LD	IY,LINBUF+1	
F11D	CD60F3	0271	CALL	PARAMS	;INPUT NUMERIC PARAMETERS FROM
F120	DDE1	0272	POP	IX	; LINE BUFFER AND TEST IF ERROR
F122	3810	0273	JR	C,WHAT-\$	
F124	2A79FF	0274	LD	HL,(PARAM1)	
F127	ED5B7BFF	0275	LD	DE,(PARAM2)	
F12B	ED4B7DFF	0276	LD	BC,(PARAM3)	
F12F	CD42F1	0277	CALL	CALLX	;CALL SUBROUTINE @ IX
F132	30B9	0278	JR	NC,PROMPT-\$;GO BACK TO PROMPT IF NO ERRORS
		0279			
F134	CDE4F3	0280	WHAT: CALL	PNEXT	
F137	20776861	0281	DEFM	' what ?'	
	74203F				
F13E	07	0282	DEFB	'G'-64	;SAY 'what ?' AND BEEP THE BELL
F13F	04	0283	DEFB	EOT	
F140	18AB	0284	JR	PROMPT-\$	
		0285 ;			
		0286 ;			
F142	DDE9	0287	CALLX: JP	(IX)	;CALL SUBROUTINE @ IX
		0288 ;			
		0289 ;			
		0290 ;			
F144	54	0291	CMDTAB: DEFB	'T'	
F145	56	0292	DEFB	'V'	
F146	52	0293	DEFB	'R'	
F147	4F	0294	DEFB	'O'	
F148	49	0295	DEFB	'I'	
F149	47	0296	DEFB	'G'	
F14A	58	0297	DEFB	'X'	
F14B	46	0298	DEFB	'F'	
F14C	4D	0299	DEFB	'M'	
F14D	43	0300	DEFB	'C'	
F14E	42	0301	DEFB	'B'	
F14F	44	0302	DEFB	'D'	
F150	41	0303	DEFB	'A'	
F151	6BF1	0304	DEFW	BOOT	;BOOT FROM DRIVE B
F153	ECF1	0305	DEFW	MEMDMP	;DUMP MEMORY IN HEX/ASCII
F155	85F1	0306	DEFW	BOOTALT	;BOOT UP CP/M
F157	D8F2	0307	DEFW	BLOCK	;MEMORY BLOCK MOVE
F159	3EF2	0308	DEFW	VIEW	;MEMORY EXAMINE/CHANGE
F15B	CAF2	0309	DEFW	FILL	;FILL MEMORY
F15D	7CF2	0310	DEFW	TEST	;RAM DIAGNOSTIC
F15F	71F2	0311	DEFW	GOTO	;JUMP TO MEMORY LOCATION
F161	06F3	0312	DEFW	INCMD	;READ FROM INPUT PORT
F163	28F3	0313	DEFW	OUTCMD	;WRITE TO OUTPUT PORT
F165	89F1	0314	DEFW	DSKCMD	;DISPLAY DISK SECTOR DATA
F167	F0F2	0315	DEFW	VERCMD	;MEMORY BLOCK COMPARE
F169	2AF4	0316	DEFW	TYPE	;TYPEWRITER MODE

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

0027      0317 ;
          0318 ;
          0319 CMDSIZ EQU    $-CMDTAB
          0320 ;
          0321 ;
          0322 ;*****
          0323 ;*
          0324 ;*      MONITOR COMMAND ACTION ROUTINES PACKAGE      *
          0325 ;*
          0326 ;*****
          0327 ;
          0328 ;
          0329 ;
          0330 ;
          0331 ;
          0332 ;      -- DISK BOOT LOADER COMMAND --
          0333 ;
F16B  OE00      0334 BOOT:  LD      C,0          ;SELECT DRIVE 0 FOR BOOT LOAD
F16D  CDDCF7    0335 BOOT1: CALL    SELECT
F170  2043      0336      JR      NZ,DSKERR-$
F172  CD12F8    0337      CALL    HOME          ;HOME HEAD TO TRACK 0
F175  203E      0338      JR      NZ,DSKERR-$    ;ERROR IF NOT READY OR AT TRO
F177  218000    0339      LD      HL,128        ;POINT TO CP/M READ BUFFER
F17A  OE01      0340      LD      C,1          ;SELECT SECTOR 1
F17C  CD5FF8    0341      CALL    READ         ;READ TRACK 0/ SECTOR 1
F17F  2034      0342      JR      NZ,DSKERR-$
F181  F1        0343      POP     AF           ;CLEAN UP STACK
F182  C38000    0344      JP      128          ;GO EXECUTE LOADER AT 128
          0345 ;
          0346 ;
          0347 ;      ALTERNATE BOOT FROM DRIVE 'B'
          0348 ;
F185  OE01      0349 BOOTALT: LD      C,1          ;LOAD THE DRIVE NUMBER
F187  18E4      0350      JR      BOOT1-$        ;CONT WITH NORMAL BOOT ROUTINE
          0351 ;
          0352 ;
          0353 ;      -- DISK SECTOR READ COMMAND --
          0354 ;
F189  FE03      0355 DSKCMD: CP      3          ;CHECK PARAMETER COUNT
F18B  37        0356      SCF
F18C  C0        0357      RET      NZ
F18D  4D        0358      LD      C,L          ;USE FIRST ARG AS UNIT#
F18E  CDDCF7    0359      CALL    SELECT
F191  2022      0360      JR      NZ,DSKERR-$
F193  217BFF    0361      LD      HL,PARAM2
F196  4E        0362      LD      C,(HL)        ;USE SECOND ARG AS TRACK#
F197  CD24F8    0363      CALL    SEEK
F19A  2019      0364      JR      NZ,DSKERR-$
F19C  217DFF    0365      LD      HL,PARAM3
F19F  4E        0366      LD      C,(HL)        ;USE THIRD ARG AS SECTOR#
F1A0  218000    0367 DSK2:  LD      HL,128
F1A3  CD5FF8    0368      CALL    READ
F1A6  CBC7      0369      SET     0,A          ;MARK ERROR BYTE AS DUE TO READ
F1A8  200B      0370      JR      NZ,DSKERR-$
F1AA  218000    0371      LD      HL,128
F1AD  110800    0372      LD      DE,8
F1B0  CDOEF2    0373      CALL    DUMP         ;DUMP DISK READ BUFFER AND
F1B3  1814      0374      JR      DSKADR-$      ; PRINT UNIT/TRACK/SECTOR
          0375 ;
F1B5  F5        0376 DSKERR: PUSH    AF          ;SAVE 1771 STATUS
F1B6  CDE4F3    0377      CALL    PNEXT
F1B9  6469736B  0378      DEFM    'disk error '
          20657272
          6F7220
F1C4  04        0379      DEFB     EOT
F1C5  F1        0380      POP     AF
F1C6  CDC8F3    0381      CALL    PUT2HS        ;PRINT ERROR STATUS IN HEX

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

F1C9 3E55      0382 DSKADR: LD      A,'U'          ;NOW DISPLAY UNIT/TRACK/SECTOR
F1CB CD0EF4    0383      CALL      OUTPUT
F1CE 3A5FFF    0384      LD        A,(UNIT)
F1D1 CDC8F3    0385      CALL      PUT2HS          ;PRINT DRIVE UNIT#
F1D4 3E54      0386      LD        A,'T'
F1D6 CD0EF4    0387      CALL      OUTPUT
F1D9 3A67FF    0388      LD        A,(TRACK)
F1DC CDC8F3    0389      CALL      PUT2HS          ;PRINT TRACK# IN HEX
F1DF 3E53      0390      LD        A,'S'
F1E1 CD0EF4    0391      CALL      OUTPUT
F1E4 3A68FF    0392      LD        A,(SECTOR)
F1E7 CDC8F3    0393      CALL      PUT2HS          ;PRINT SECTOR# IN HEX
F1EA B7        0394      OR        A
F1EB C9        0395      RET
                0396 ;
                0397 ;
                0398 ;
                0399 ;
                0400 ;      -- MEMORY DUMP COMMAND --
                0401 ;
F1EC 3D        0402 MEMDMP: DEC      A          ;CHECK PARAMETER COUNT
F1ED 2806      0403      JR        Z,MDMP2-$
F1EF 3D        0404      DEC      A
F1F0 2808      0405      JR        Z,MDMP3-$
F1F2 2A82FF    0406 MDMP1: LD      HL,(LAST)
F1F5 111000    0407 MDMP2: LD      DE,16
F1F8 180D      0408      JR        MDMP3B-$
                0409
F1FA EB        0410 MDMP3: EX      DE,HL
F1FB ED52      0411      SBC      HL,DE          ;DERRIVE BYTECOUNT FOR DUMP RANGE
F1FD 0604      0412      LD      B,4
F1FF CB3C      0413 MDMP3A: SRL     H          ;DIVIDE BYTECOUNT BY 16
F201 CB1D      0414      RR
F203 10FA      0415      DJNZ     MDMP3A-$
F205 23        0416      INC      HL
F206 EB        0417      EX      DE,HL
F207 CD0EF2    0418 MDMP3B: CALL    DUMP          ;DUMP DE*16 BYTES STRTING AT HL
F20A 2282FF    0419      LD      (LAST),HL
F20D C9        0420      RET
                0421 ;
                0422 ;
F20E E5        0423 DUMP:  PUSH     HL          ;SAVE STARTING ADDRESS
F20F CDC3F3    0424      CALL      PUT4HS          ;PRINT STARTING ADDRESS IN HEX
F212 CDFAF3    0425      CALL      SPACE
F215 0610      0426      LD      B,16
F217 7E        0427 DUMP2: LD      A,(HL)          ;GET A DATA BYTE @ HL
F218 23        0428      INC      HL
F219 CDC8F3    0429      CALL      PUT2HS          ;PRINT THE DATA IN HEX
F21C 10F9      0430      DJNZ     DUMP2-$          ;REPEAT 16 TIMES
F21E E1        0431      POP      HL          ;RESTORE STARTING ADDRESS
F21F 0610      0432      LD      B,16
F221 7E        0433 DUMP3: LD      A,(HL)          ;GET BACK DATA BYTE @ HL
F222 23        0434      INC      HL
F223 CBBF      0435      RES      7,A
F225 FE20      0436      CP      20H
F227 3804      0437      JR      C,DUMP4-$
F229 FE7F      0438      CP      7FH
F22B 3802      0439      JR      C,DUMP5-$
F22D 3E2E      0440 DUMP4: LD      A,'.'          ;PRINT A DOT IF DATA 20 OR 7F
F22F CD0EF4    0441 DUMP5: CALL      OUTPUT          ;PRINT ASCII CHARACTER IN A
F232 10ED      0442      DJNZ     DUMP3-$
F234 CDF4F3    0443      CALL      CRLFS
F237 C0        0444      RET      NZ          ;EXIT IF ESC REQST IS INDICATED
F238 1B        0445      DEC      DE
F239 7A        0446      LD      A,D
F23A B3        0447      OR      E
F23B 20D1      0448      JR      NZ,DUMP-$

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

F23D C9      0449      RET
              0450 ;
              0451 ;
              0452 ;
              0453 ;
              0454 ;      -- MEMORY EXAMINE COMMAND --
              0455 ;

F23E CDBFF2  0456 VIEW: CALL  MDATA
F241 CD00F4  0457      CALL  ECHO
F244 FE0D    0458      CP     CR
F246 2824    0459      JR     Z,VIEW4-$
F248 FE2D    0460      CP     '-'
F24A 2822    0461      JR     Z,VIEW5-$
F24C FE2C    0462      CP     ','
F24E 2005    0463      JR     NZ,VIEW2-$
F250 CD00F4  0464      CALL  ECHO
F253 1813    0465      JR     VIEW3-$
              0466

F255 CDB3F3  0467 VIEW2: CALL  ASCHEX
F258 3F      0468      CCF
F259 D0      0469      RET     NC
F25A 07      0470      RLCA
F25B 07      0471      RLCA
F25C 07      0472      RLCA
F25D 07      0473      RLCA
F25E 4F      0474      LD     C,A
F25F CD00F4  0475      CALL  ECHO
F262 CDB3F3  0476      CALL  ASCHEX
F265 3F      0477      CCF
F266 D0      0478      RET     NC
F267 B1      0479      OR     C
F268 77      0480 VIEW3: LD     (HL),A
F269 CDA9F2  0481      CALL  CHECK
F26C 23      0482 VIEW4: INC    HL
F26D 23      0483      INC    HL
F26E 2B      0484 VIEW5: DEC    HL
F26F 18CD    0485      JR     VIEW-$
              0486 ;
              0487 ;
              0488 ;
              0489 ;      -- JUMP TO MEMORY LOCATION COMMAND --
              0490 ;

F271 3D      0491 GOTO:  DEC    A      ;CHECK PARAMETER COUNT
F272 37      0492      SCF
F273 C0      0493      RET     NZ
F274 E5      0494      PUSH   HL
F275 DDE1    0495      POP     IX
F277 CD42F1  0496      CALL  CALLX      ;CALL ADDRESS PASSED IN HL
F27A B7      0497      OR     A
F27B C9      0498      RET          ;RETURN IF WE GET BACK AGAIN
              0499 ;
              0500 ;
              0501 ;
              0502 ;      -- MEMORY READ/WRITE DIAGNOSTIC COMMAND --
              0503 ;

F27C FE02    0504 TEST:  CP     2      ;CHECK PARAMETER COUNT
F27E 37      0505      SCF
F27F C0      0506      RET     NZ
F280 13      0507      INC    DE
F281 5A      0508      LD     E,D      ;GET ENDING PAGE ADDRESS INTO E
F282 54      0509      LD     D,H      ;GET STARTING PAGE ADDRESS INTO D
F283 0600    0510      LD     B,0      ;INITIALIZE PASS COUNTER
F285 62      0511 TEST1: LD     H,D      ;POINT HL TO START OF BLOCK
F286 2E00    0512      LD     L,0
F288 7D      0513 TEST2: LD     A,L
F289 AC      0514      XOR     H      ;GENERATE TEST BYTE
F28A A8      0515      XOR     B

```


F28B	77	0516	LD	(HL),A	;STORE BYTE IN RAM
F28C	23	0517	INC	HL	
F28D	7C	0518	LD	A,H	
F28E	BB	0519	CP	E	;CHECK FOR END OF TEST BLOCK
F28F	20F7	0520	JR	NZ,TEST2-\$	
		0521			NOW READ BACK EACH BYTE & COMPARE
		0522	LD	H,D	
F291	62	0522	LD	L,0	;POINT HL BACK TO START
F292	2E00	0523	LD	A,L	
F294	7D	0524	LD	A,L	
F295	AC	0525	XOR	H	;RE-GENERATE TEST BYTE DATA
F296	A8	0526	XOR	B	
F297	CDA9F2	0527	CALL	CHECK	;VERIFY MEMORY DATA STILL GOOD
F29A	C0	0528	RET	NZ	;EXIT IF ESC REQST IS INDICATED
F29B	23	0529	INC	HL	; ELSE GO ON TO NEXT BYTE
F29C	7C	0530	LD	A,H	
F29D	BB	0531	CP	E	;CHECK FOR END OF BLOCK
F29E	20F4	0532	JR	NZ,TEST3-\$	
F2A0	04	0533	INC	B	;BUMP PASS COUNT
F2A1	3E2B	0534	LD	A,'+'	
F2A3	CDOEF4	0535	CALL	OUTPUT	;PRINT '+' AND ALLOW FOR EXIT
F2A6	28DD	0536	JR	Z,TEST1-\$;DO ANOTHER PASS IF NO ESCAPE
F2A8	C9	0537	RET		
		0538			
		0539			
		0540			
F2A9	BE	0541	CHECK: CP	(HL)	
F2AA	C8	0542	RET	Z	;RETURN IF (HL)=A
F2AB	F5	0543	PUSH	AF	
F2AC	CDBFF2	0544	CALL	MDATA	;PRINT WHAT WAS ACTUALLY READ
F2AF	CDE4F3	0545	CALL	PNEXT	
F2B2	73686F75	0546	DEFM	'should='	
	6C643D				
F2B9	04	0547	DEFB	EOT	
F2BA	F1	0548	POP	AF	
F2BB	CDC8F3	0549	CALL	PUT2HS	;PRINT WHAT SHOULD HAVE BEEN READ
F2BE	C9	0550	RET		
		0551			
		0552			
F2BF	CDF4F3	0553	MDATA: CALL	CRLFS	
F2C2	CDC3F3	0554	CALL	PUT4HS	
F2C5	7E	0555	LD	A,(HL)	
F2C6	CDC8F3	0556	CALL	PUT2HS	
F2C9	C9	0557	RET		
		0558			
		0559			
		0560			
		0561			
		0562			
		0563	FILL: CP	3	;CHECK IF PARAMETER COUNT=3
F2CA	FE03	0564	SCF		
F2CC	37	0565	RET	NZ	
F2CD	C0	0566	LD	(HL),C	
F2CE	71	0567	PUSH	HL	
F2CF	E5	0568	OR	A	
F2D0	B7	0569	SBC	HL,DE	;COMPARE HL TO END ADDRESS IN DE
F2D1	ED52	0570	POP	HL	
F2D3	E1	0571	INC	HL	;ADVANCE POINTER AFTER COMPARISON
F2D4	23	0572	JR	C,FILL1-\$	
F2D5	38F7	0573	RET		
F2D7	C9	0574			
		0575			
		0576			
		0577			
		0578			
		0579			
		0580	BLOCK: CP	3	;CHECK IF PARAMETER COUNT=3
F2D8	FE03	0581	SCF		
F2DA	37				

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F2DB	C0	0582	RET	NZ	
F2DC	CDE5F2	0583	CALL	BLOCAD	
F2DF	79	0584	LD	A,C	
F2E0	B0	0585	OR	B	
F2E1	C8	0586	RET	Z	;EXIT NOW IF BC=0
F2E2	EDB0	0587	LDIR		
F2E4	C9	0588	RET		
		0589 ;			
		0590 ;			
		0591 ;			
F2E5	EB	0592	BLOCAD: EX	DE,HL	
F2E6	B7	0593	OR	A	;CLEAR CARRY
F2E7	ED52	0594	SBC	HL,DE	;GET DIFFERENCE BETWEEN
F2E9	EB	0595	EX	DE,HL	;HL & DE FOR BYTECOUNT
F2EA	D5	0596	PUSH	DE	
F2EB	C5	0597	PUSH	BC	
F2EC	D1	0598	POP	DE	;GET OLD BC INTO DE
F2ED	C1	0599	POP	BC	
F2EE	03	0600	INC	BC	;GET COUNT+1 INTO BC
F2EF	C9	0601	RET		
		0602 ;			
		0603 ;			
		0604 ;			
		0605 ;			
		0606 ;			
			-- MEMORY BLOCK COMPARE COMMAND --		
F2F0	FE03	0607	VERCMD: CP	3	;CHECK IF PARAMETER COUNT=3
F2F2	37	0608	SCF		
F2F3	C0	0609	RET	NZ	
F2F4	CDE5F2	0610	CALL	BLOCAD	
F2F7	1808	0611	JR	VERF2-\$	
		0612			
F2F9	1A	0613	VERF1: LD	A,(DE)	
F2FA	CDA9F2	0614	CALL	CHECK	;COMPARE DATA @ DE AND @ HL
F2FD	C0	0615	RET	NZ	;EXIT IF ESCAPE REQ IS INDICATED
F2FE	23	0616	INC	HL	
F2FF	13	0617	INC	DE	
F300	0B	0618	DEC	BC	
F301	78	0619	VERF2: LD	A,B	
F302	B1	0620	OR	C	
F303	20F4	0621	JR	NZ,VERF1-\$	
F305	C9	0622	RET		
		0623 ;			
		0624 ;			
		0625 ;			
		0626 ;			
		0627 ;			
		0628 ;			
			-- READ FROM INPUT PORT COMMAND --		
F306	3D	0629	INCMD: DEC	A	;CHECK IF PARAMETER COUNT=1
F307	37	0630	SCF		
F308	C0	0631	RET	NZ	
F309	4D	0632	LD	C,L	;POINT C TO INPUT PORT
F30A	CDF4F3	0633	IN1: CALL	CRLFS	
F30D	79	0634	LD	A,C	
F30E	CDC8F3	0635	CALL	PUT2HS	
F311	ED78	0636	IN	A,(C)	
F313	CDC8F3	0637	CALL	PUT2HS	
F316	CD00F4	0638	CALL	ECHO	
F319	FE0D	0639	CP	CR	
F31B	2806	0640	JR	Z,IN2-\$	
F31D	FE2D	0641	CP	'-'	
F31F	2804	0642	JR	Z,IN3-\$	
F321	B7	0643	OR	A	
F322	C9	0644	RET		
		0645			
F323	0C	0646	IN2: INC	C	
F324	0C	0647	INC	C	
F325	0D	0648	IN3: DEC	C	

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

F326 18E2      0649      JR      IN1-$
                0650 ;
                0651 ;
                0652 ;
                0653 ;      -- WRITE TO OUTPUT PORT COMMAND --
                0654 ;
F328 FE02      0655 OUTCMD: CP      2      ;CHECK IF PARAMETER COUNT=2
F32A 37        0656      SCF
F32B C0        0657      RET      NZ
F32C 4D        0658      LD      C,L      ;POINT C TO OUTPUT PORT
F32D ED59      0659      OUT      (C),E    ;OUTPUT DATA PASSED IN E
F32F B7        0660      OR      A
F330 C9        0661      RET
                0662 ;
                0663 ;
                0664 ;*****
                0665 ;*
                0666 ;*      CONSOLE I/O PACKAGE AND UTILITY ROUTINES      *
                0667 ;*
                0668 ;*****
                0669 ;
                0670 ;
                0671 ;
F331 41        0672 GETLIN: LD      B,C      ;SAVE MAX LINE LNTH PARAMTR IN B
F332 CD00F4    0673 GLIN1: CALL    ECHO      ;GET A CHARACTER FROM THE CONSOLE
F335 FE0D      0674      CP      CR      ;CHECK FOR CARRIAGE RETURN
F337 280E      0675      JR      Z,GLIN2-$
F339 FE08      0676      CP      'H'-64      ;CHECK FOR CTL-H BACKSPACE
F33B 280C      0677      JR      Z,GLIN4-$
F33D FE20      0678      CP      ' '
F33F D8        0679      RET      C      ;OTHER CONTROL CHARS ARE ILLEGAL
F340 77        0680      LD      (HL),A
F341 23        0681      INC      HL      ;STORE CHARACTER IN BUFFER
F342 0D        0682      DEC      C
F343 20ED      0683      JR      NZ,GLIN1-$    ;GET ANOTHER IF THERE'S MORE ROOM
F345 37        0684      SCF
F346 C9        0685      RET
                0686      ;RETURN WITH CARRY=1 IF TOO
                0687      ;MANY CHARACTERS ARE ENTERED
                0688      ;PUT CARRIAGE RET ON END OF LINE
                0689      ;RETURN WITH CARRY BIT=0
F347 77        0687 GLIN2: LD      (HL),A
F348 C9        0688      RET
                0689
F349 2B        0690 GLIN4: DEC      HL      ;DELETE LAST CHAR FROM BUFFER
F34A CDE4F3    0691      CALL    PNEXT
F34D 2008      0692      DEFB      ' ','H'-64    ;PRINT A SPACE TO OVERWRITE THE
F34F 04        0693      DEFB      EOT      ; LAST CHAR, THEN DO A BACKSPACE
F350 0C        0694      INC      C
F351 78        0695      LD      A,B      ;MAKE SURE YOU'RE NOT TRYING TO
F352 91        0696      SUB      C      ;BACKSP PAST THE START OF THE LINE
F353 30DD      0697      JR      NC,GLIN1-$
F355 C9        0698      RET
                0699 ;
                0700 ;
                0701 ;
F356 EDB1      0702 SEARCH: CPIR      ;SEARCH TABLE @HL FOR MATCH WITH A
F358 C0        0703      RET      NZ      ;EXIT NOW IF SEARCH FAILS
F359 09        0704      ADD      HL,BC
F35A 09        0705      ADD      HL,BC      ;ADD RESIDUE FROM CPIR BYTECOUNT
F35B 09        0706      ADD      HL,BC      ; TO HL 3 TIMES TO GET POINTER
F35C 4E        0707      LD      C,(HL)    ; TO ADDRESS PART OF TABLE ENTRY
F35D 23        0708      INC      HL
F35E 46        0709      LD      B,(HL)
F35F C9        0710      RET
                0711 ;
                0712 ;
                0713 ;
                0714 ;
F360 010000    0715 PARAMS: LD      BC,0

```

F363	FD7E00	0716	LD	A,(IY+0)	
F366	FE0D	0717	CP	CR	;CHECK IF LINE TERMINATES
F368	2008	0718	JR	NZ,PARA2-\$; IMMEDIATELY WITH A RETURN
F36A	AF	0719	XOR	A	
F36B	C9	0720	RET		;RETURN WITH PARAM COUNT=0 IF SO
		0721			
F36C	0C	0722	PARA1: INC	C	
F36D	0C	0723	INC	C	
F36E	CB59	0724	BIT	3,C	
F370	37	0725	SCF		
F371	C0	0726	RET	NZ	;ERROR IF 4 NUMBERS ENTERED
F372	C5	0727	PARA2: PUSH	BC	;SAVE PARAMETER COUNT
F373	CD95F3	0728	CALL	GETHEX	;READ A NUMBER FROM LINE BUFFER
F376	C1	0729	POP	BC	
F377	D8	0730	PARA4: RET	C	;ERROR IF RESULT OVER 16 BITS
F378	DD2179FF	0731	LD	IX,PARAM1	;POINT TO PARAMETER STORAGE AREA
F37C	DD09	0732	ADD	IX,BC	;ADD PARAMETER COUNT IN BC
F37E	DD7500	0733	LD	(IX+0),L	
F381	DD7401	0734	LD	(IX+1),H	;STORE DATA RETURNED FROM 'GETHEX'
F384	FE20	0735	CP	' '	
F386	28E4	0736	JR	Z,PARA1-\$;GET ANOTHER ITEM IF SPACE
F388	FE2C	0737	CP	' , '	
F38A	28E0	0738	JR	Z,PARA1-\$;GET ANOTHER ITEM IF COMMA
F38C	FE0D	0739	CP	CR	
F38E	37	0740	SCF		;ELSE CHECK FOR CARRIAGE RETURN
F38F	C0	0741	RET	NZ	; AND EXIT WITH CY=1 IF NOT
F390	79	0742	PAREND: LD	A,C	
F391	CB3F	0743	SRL	A	;A=COUNT OF NUMBERS ENTERED
F393	3C	0744	INC	A	
F394	C9	0745	RET		
		0746 ;			
		0747 ;			GETHEX CONVERTS ASCII TO BINARY AND DOES
		0748 ;			HIGH LIMIT CHECKS TO LESS THAN 17 BITS.
		0749 ;			CARRY SET ON ILLEGAL CONVERSION RESULT
		0750 ;			TERMINATING CHARACTER RETURNS IN A.
		0751 ;			HL RETURNS WITH 16 BIT BINARY INTEGER
		0752 ;			
F395	210000	0753	GETHEX: LD	HL,0	
F398	180B	0754	JR	GNUM3-\$	
		0755			
F39A	0604	0756	GNUM1: LD	B,4	
F39C	29	0757	GNUM2: ADD	HL,HL	;MULTIPLY RESULT BY 16
F39D	D8	0758	RET	C	;RETURN IF IT OVERFLOWS 16 BITS
F39E	10FC	0759	DJNZ	GNUM2-\$	
F3A0	5F	0760	LD	E,A	;APPEND NEW LOW ORDER DIGIT
F3A1	1600	0761	LD	D,0	;AND GET RESULT BACK INTO DE
F3A3	19	0762	ADD	HL,DE	
F3A4	D8	0763	RET	C	;RETURN IF OVERFLOW
F3A5	FD7E00	0764	GNUM3: LD	A,(IY+0)	;GET A CHARACTER FROM LINE INPUT
F3A8	FD23	0765	INC	IY	; BUFFER @ IY AND BUMP IY
F3AA	4F	0766	LD	C,A	
F3AB	CDB3F3	0767	CALL	ASCHEX	;CONVERT ASCII TO NUMERIC
F3AE	30EA	0768	JR	NC,GNUM1-\$	
F3B0	79	0769	LD	A,C	
F3B1	B7	0770	OR	A	
F3B2	C9	0771	RET		
		0772 ;			
		0773 ;			
F3B3	D630	0774	ASCHEX: SUB	'0'	
F3B5	D8	0775	RET	C	
F3B6	FE0A	0776	CP	10	
F3B8	3F	0777	CCF		
F3B9	D0	0778	RET	NC	
F3BA	D607	0779	SUB	7	
F3BC	FE0A	0780	CP	10	
F3BE	D8	0781	RET	C	
F3BF	FE10	0782	CP	16	

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F3C1	3F	0783	CCF	
F3C2	C9	0784	RET	
		0785	;	
		0786	;	
		0787	;	
F3C3	7C	0788	PUT4HS: LD	A,H
F3C4	CDCFF3	0789	CALL	PUT2HX
F3C7	7D	0790	LD	A,L
F3C8	CDCFF3	0791	PUT2HS: CALL	PUT2HX
F3CB	CDFAF3	0792	CALL	SPACE
F3CE	C9	0793	RET	
		0794	;	
		0795	;	
F3CF	F5	0796	PUT2HX: PUSH	AF
F3D0	1F	0797	RRA	
F3D1	1F	0798	RRA	
F3D2	1F	0799	RRA	
F3D3	1F	0800	RRA	
F3D4	CDD8F3	0801	CALL	PUTNIB
F3D7	F1	0802	POP	AF
F3D8	E60F	0803	PUTNIB: AND	00001111B
F3DA	C690	0804	ADD	A,90H
F3DC	27	0805	DAA	
F3DD	CE40	0806	ADC	A,40H
F3DF	27	0807	DAA	
F3E0	CDOEF4	0808	CALL	OUTPUT
F3E3	C9	0809	RET	
		0810	;	
		0811	;	
		0812	;	PMSG PRINTS THE STRING OF ASCII CHARACTERS
		0813	;	POINTED TO BY THE RELATIVE ADDRESS IN DE
		0814	;	UNTIL AN EOT IS ENCOUNTERED IN THE STRING.
		0815	;	
0004		0816	EOT EQU	04H
000D		0817	CR EQU	0DH
000A		0818	LF EQU	0AH
		0819	;	
		0820		
F3E4	E3	0821	PNEXT: EX	(SP),HL
F3E5	CDEAF3	0822	CALL	PMSG
F3E8	E3	0823	EX	(SP),HL
F3E9	C9	0824	RET	
		0825	;	
F3EA	7E	0826	PMSG: LD	A,(HL)
F3EB	23	0827	INC	HL
F3EC	FE04	0828	CP	EOT
F3EE	C8	0829	RET	Z
F3EF	CDOEF4	0830	CALL	OUTPUT
F3F2	18F6	0831	JR	PMSG-\$
		0832	;	
		0833	;	
		0834	;	CRLFS OUTPUTS A RETURN-LINEFEED-SPACE
		0835	;	TO THE CONSOLE DEVICE
		0836	;	
F3F4	CDE4F3	0837	CRLFS: CALL	PNEXT
F3F7	OD0A04	0838	DEFB	CR,LF,EOT
F3FA	3E20	0839	SPACE: LD	A,' '
F3FC	CDOEF4	0840	CALL	OUTPUT
F3FF	C9	0841	RET	
		0842	;	
		0843	;	
		0844	;	
		0845	;	ECHO INPUTS ONE CHARACTER FROM THE CONSOLE
		0846	;	DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND
		0847	;	THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET
		0848	;	
		0849	;	OUTPUT PRINTS THE CHARACTER IN REGISTER A ON

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

0850 ; THE CONSOLE OUTPUT DEVICE AND THEN DOES A CHECK
0851 ; FOR CONSOLE INPUT TO FREEZE OR ABORT OUTPUT.
0852 ;
0853
F400 CD09F0 0854 ECHO: CALL CONIN ;INPUT A CHARACTER AND ECHO IT
F403 F5 0855 PUSH AF
F404 CD0CF0 0856 CALL CONOUT
F407 F1 0857 POP AF
F408 FE5B 0858 CP 'Z'+1
F40A D8 0859 RET C
F40B D620 0860 SUB 32 ;CONVERT UPPER CASE TO LOWER CASE
F40D C9 0861 RET
0862 ;
0863 ;
0864 ;
F40E CD0CF0 0865 OUTPUT: CALL CONOUT
F411 CD06F0 0866 CALL CONST ;SEE IF CONSOLE INPUT IS PENDING
F414 280F 0867 JR Z,OUTP2-$
F416 CD09F0 0868 CALL CONIN
F419 FE0D 0869 CP CR ;SEE IF CARRIAGE RETRN WAS TYPED
F41B 2805 0870 JR Z,OUTP1-$
F41D CD09F0 0871 CALL CONIN ;WAIT FOR ANOTHER INPUT CHAR
F420 1803 0872 JR OUTP2-$ ; THEN RETURN TO CALLING ROUTINE
0873
F422 3281FF 0874 OUTP1: LD (ESCFLG),A ;SET ESC FLAG TO NON-ZERO VALUE
F425 3A81FF 0875 OUTP2: LD A,(ESCFLG)
F428 B7 0876 OR A ;RETURN CURRENT STATUS OF ESCAPE
F429 C9 0877 RET ; FLAG TO CALLING ROUTINE
0878 ;
0879 ;
0880 ;
0881 INCLUDE TYPE.ASM
0882 *****
0883 ;*
0884 ;*
0885 ;* XEROX 820 TYPEWRITER MODE *
0886 ;* *
0887 ;*****
0888 ;
0889 ;
F42A 0890 TYPE: ORG $
F42A 7D 0891 LD A,L ;GET BAUD RATE IN L
F42B E60F 0892 AND OFH ;USE VALUES FROM 0 TO 15
F42D 2002 0893 JR NZ,BAUD-$ ;DEFLT ZERO FOR 1200 BAUD
F42F 3E07 0894 LD A,7
F431 0895 BAUD:
F431 D30C 0896 OUT (OCH),A ;SET UP BAUD RATE FOR CH B
F433 3E1A 0897 LD A,01AH ;CLR SCRN TO CURSOR TO LEFT
F435 CD4BF6 0898 CALL CRTOUT
F438 CDE4F3 0899 CALL PNEXT ;DISPLAY THE FLWNG MESSAGES
F43B 0900 MESS:
F43B 2E2E2E38 0901 DEFM '...820 TYPEWRITER VER. 1.0...'
32302054
59504557
52495445
52202056
45522E20
312E302E
2E2E
F459 ODOA 0902 DEFB ODH,0AH ;CR,LF
F45B 20202050 0903 DEFM ' PRESS CTRL+X TO EXIT'
52455353
20435452
4C2B5820
544F2045
584954
F472 ODOA 0904 DEFB ODH,0AH ;CR,LF

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F474	04	0905	DEFB	04H	;END OF TEXT
F475	2152F5	0906	LD	HL,PRTINI	;GET PRT INIT COMMANDS
F478	0609	0907	LD	B,9	;GET COMMAND COUNT
F47A	CD4AF5	0908	CALL	INILUP	;RESET PRINTER
F47D	0E05	0909	LD	C,5	;SET COUNTER OF 5 SPACES
F47F	1619	0910	LD	D,25	;SET COUNTER FOR 25 TABS
F481	79	0911	LD	A,C	
F482		0912	TABSET:		
F482	3267F5	0913	LD	(TBCMD+7),A	;SAVE TAB POSITION
F485	2160F5	0914	LD	HL,TBCMD	;SEND TAB COMMAND TO PRT
F488	060F	0915	LD	B,15	;SEND ABS TAB AND SET TAB
F48A	CD4AF5	0916	CALL	INILUP	
F48D	3E05	0917	LD	A,5	;SET UP NEXT TAB POSITION
F48F	81	0918	ADD	A,C	
F490	4F	0919	LD	C,A	;AND SAVE IT
F491	15	0920	DEC	D	
F492	20EE	0921	JR	NZ,TABSET-\$;UNTIL 25 TABS ARE SET
F494	3E0D	0922	LD	A,ODH	
F496	CD40F6	0923	CALL	SIOOUT	;AND SEND CR
		0924	;SET UP LEFT MARGIN AT 12		
F499	215BF5	0925	LD	HL,LMTAB	;SET UP COMMAND TABLE FOR
		0926			;LEFT MARGIN
F49C	0605	0927	LD	B,5	;SEND CARRIAGE TO COL 12
F49E	CD4AF5	0928	CALL	INILUP	;AND SET LEFT MARGIN THERE
F4A1	3E0C	0929	LD	A,12	;INIT MARGIN AND COL COUNT
F4A3	2178F5	0930	LD	HL,LPLC	
F4A6	77	0931	LD	(HL),A	
F4A7	23	0932	INC	HL	
F4A8	77	0933	LD	(HL),A	
F4A9	AF	0934	XOR	A	
F4AA	23	0935	INC	HL	
F4AB	77	0936	LD	(HL),A	;RESET ESCAPE SEQUENCE
F4AC		0937	TYPLUP:		
F4AC	CD06F0	0938	CALL	CONST ;KEY IN	INPUT BUFFER?
F4AF	28FB	0939	JR	Z,TYPLUP-\$;WAIT UNTIL KEY IN INPUT BFR
F4B1	3A7AF5	0940	LD	A,(ESCKEY)	
F4B4	D601	0941	SUB	1	
F4B6	CE00	0942	ADC	A,0	;DECRSE ESC COUNTER UNTL ZERO
F4B8	327AF5	0943	LD	(ESCKEY),A	
F4BB		0944	KEYIN:		
		0945	;		
F4BB	CD09F0	0946	CALL	CONIN	;GET KEY IN INPUT BUFFER
F4BE	2178F5	0947	LD	HL,LPLC	;GET PRT COL COUNTER ADDRS
F4C1	4F	0948	LD	C,A	;SAVE KEY IN REGISTER C
F4C2	FE20	0949	CP	020H	;PRINTABLE CHARACTER?
F4C4	D226F5	0950	JP	NC,PRTKEY	;YES PRINTABLE CHARACTER
F4C7		0951	CNTKEY:		
F4C7	FE0D	0952	CP	ODH	;KEY IS CR?
F4C9	200F	0953	JR	NZ,NOCR-\$;NOT A CR
F4CB		0954	CARET:		
F4CB	3A79F5	0955	LD	A,(LFMG)	;GET LEFT MARGIN
F4CE	77	0956	LD	(HL),A	;SET PRT COL COUNT TO LFT MRGN
F4CF	216FF5	0957	LD	HL,CRLF	;SEND CR AND LF TO PRT
F4D2	0609	0958	LD	B,9	
F4D4	CD4AF5	0959	CALL	INILUP	
F4D7	C3ACF4	0960	JP	TYPLUP	;AND GET ANOTHER KEY
F4DA		0961	NOCR:		
		0962	;		
F4DA	FE18	0963	CP	18H	;KEY IS CNTR-X?
F4DC	C2EAF4	0964	JP	NZ,NOX	;NO, TEST FOR OTHER KEY
F4DF	216FF5	0965	LD	HL,CRLF	;SEND CRLF TO PRINTER
F4E2	0609	0966	LD	B,9	
F4E4	CD4AF5	0967	CALL	INILUP	
F4E7	C300F0	0968	JP	COLD	
F4EA		0969	NOX:		
		0970	;		
F4EA	FE1B	0971	CP	01BH	;KEY IS ESC KEY?

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F4EC	2008	0972	JR	NZ,NOESC-\$;NOT AN ESCAPE KEY
		0973 ;			
		0974 ;ESCAPE KEY PRESSED			
		0975 ;			
F4EE	3E03	0976	LD	A,3	;SET UP 3 BYTE ESC KEY SEQ
F4F0	327AF5	0977	LD	(ESCKEY),A	
F4F3	C343F5	0978	JP	PRTOUT	;SND ESC KEY TO PRT AND GET
		0979 ;			;ANOTHER KEY
		0980 ;			
		0981 ;NOT AN ESCAPE KEY			
		0982 ;			
F4F6		0983 NOESC:			
		0984 ;			
F4F6	FE09	0985	CP	09H	;KEY IS TAB KEY?
F4F8	201B	0986	JR	NZ,NOTAB-\$;NOT A TAB KEY
		0987 ;			
		0988 ;TAB KEY PRESSED			
		0989 ;			
		0990 ;COMPARE CURRENT PRT COLUMN POSITION WITH LIST OF TAB COLUMN			
		0991 ;AND USE THE NEXT LARGER VALUE OF TAB POSITION TO BE			
		0992 ;CURRENT POSITION			
		0993 ;			
F4FA	DD217BF5	0994	LD	IX,TABTBL	;SET UP ADDRS OF TAB TBL
F4FE	46	0995	LD	B,(HL)	;SET UP CURRENT PRT PSTN
		0996 ;			
F4FF		0997 TBLUP:			
		0998 ;			
F4FF	DD7E00	0999	LD	A,(IX)	;GET TAB COLUMN NUMBER
F502	A7	1000	AND	A	;TAB COLUMN IS ZERO?
F503	280B	1001	JR	Z,COL132-\$;ERROR, TAB NOT FOUND
F505	DD23	1002	INC	IX	;GET NEXT ADDRS OF TAB COL
F507	B8	1003	CP	B	;COMP WITH CURRENT PRT PSTN
F508	38F5	1004	JR	C,TBLUP-\$;UNTIL TAB COL NUMBER IS
F50A	28F3	1005	JR	Z,TBLUP-\$;GREATER
F50C	77	1006	LD	(HL),A	;THEN USE IT AS CURRENT COL
F50D	C343F5	1007	JP	PRTOUT	;AND SND TAB KEY OUT TO PRT
		1008 ;			
		1009 ;			
		1010 ;PRINT BELL TO INDICATE AT RIGHT MARGIN ON THE PRINTER			
		1011 ;			
		1012 ;			
		1013 ;			
F510		1014 COL132:			
F510		1015 COLO:			
		1016 ;			
F510	OE07	1017	LD	C,07H	;PRINT BELL
F512	C343F5	1018	JP	PRTOUT	;AND GET ANOTHER KEY
		1019 ;			
		1020 ;			
		1021 ;NOT A TAB KEY			
		1022 ;			
F515		1023 NOTAB:			
		1024 ;			
F515	FE08	1025	CP	08H	;KEY IS BACK SPACE KEY?
F517	202A	1026	JR	NZ,PRTOUT-\$;NOT A BACK SPACE KEY
		1027 ;			
		1028 ;BACK SPACE KEY PRESSED			
		1029 ;			
		1030			
		1031			
F519	3A79F5	1032	LD	A,(LFMG)	;GET LEFT MARGIN IN B
F51C	47	1033	LD	B,A	
F51D	7E	1034	LD	A,(HL)	;GET PRINTER COLUMN COUNT
F51E	B8	1035	CP	B	;AT LEFT MARGIN?
F51F	CA10F5	1036	JP	Z,COLO	;YES, PRINT BELL
		1037 ;			
		1038 ;			

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)


```

1039 ;
1040 ;
F522 35 1041 DEC (HL) ;DECREASE PRT COLUMN COUNT
F523 C343F5 1042 JP PRTOUT ;PRINT BACK SPACE
1043 ;
1044 ;PRINTABLE CHARACTER
1045 ;
F526 1046 PRTKEY: ;
1047 ;
F526 7E 1048 LD A,(HL) ;GET PRT COLUMN COUNT
F527 FE84 1049 CP 132 ;REACH RIGHT MARGIN?
F529 CA10F5 1050 JP Z,COL132 ;YES, PRINT BELL
F52C 3A7AF5 1051 LD A,(ESCKEY) ;KEY IS WITHIN ESC SEQ?
F52F A7 1052 AND A
F530 280D 1053 JR Z,INCCOL-$ ;NO, PRNT CHAR WITH INCRS
F532 79 1054 LD A,C ;GET CHARACTER
F533 FE39 1055 CP 039H ;CHAR IS NUMBER 9?
F535 C243F5 1056 JP NZ,PRTOUT ;NO,JUST SEND CHAR TO PRT
1057 ;
1058 ;SET NEW LEFT MARGIN
1059 ;
F538 7E 1060 LD A,(HL) ;GET CURRENT COLUMN COUNT
F539 3279F5 1061 LD (LFMG),A ;AS LEFT MARGIN
F53C C343F5 1062 JP PRTOUT ;SEND CHAR TO PRT
1063 ;
F53F 1064 INCCOL: ;
1065 ;
1066 ;INCREASE COLUMN COUNTER
1067 ;
F53F 34 1068 INC (HL) ;INC PRT COL COUNTER
F540 C343F5 1069 JP PRTOUT ;PRT CHAR & GET ANTher KEY
1070 ;
F543 1071 PRTOUT: ;
F543 79 1072 LD A,C ;GET PRINT CHARACTER
F544 CD40F6 1073 CALL SIOOUT ;SEND IT TO USART PORT B
F547 C3ACF4 1074 JP TYPLUP ;GET ANOTHER KEY
1075 ;
F54A 1076 INILUP ;
1077 ;
F54A 7E 1078 LD A,(HL) ;GET COMMAND
F54B CD40F6 1079 CALL SIOOUT ;SEND IT TO SIO PORT B
F54E 23 1080 INC HL
F54F 10F9 1081 DJNZ INILUP-$ ;UNTIL B BYTES ARE SENT
F551 C9 1082 RET
1083 ;
1084 ;
1085 ;*****
1086 ;*
1087 ;* TYPEWRITER MODE DATA BASE
1088 ;*
1089 ;*****
1090 ;
1091 ;PRINTER INITIALIZATION COMMANDS
1092 ;PRINTER RESET COMMAND
1093 ;12 SPACES
1094 ;SET LEFT MARGIN TO COLUMN 12
1095 ;
F552 1B0D50 1096 PRTINI: DEFB 01BH,0DH,050H ;ESC CR P SEQUENCE
F555 00000000 1097 DEFB 0,0,0,0,0,0
0000
F55B 1B090C 1098 LMTAB: DEFB 1BH,09H,0CH ;TAB TO COLUMN 12
F55E 1B39 1099 DEFB 1BH,39H ;SET LEFT MARGIN
1100 ;
1101 ;SET TAB AT EVERY 5 COLUMN
1102 ;
F560 00000000 1103 TBCMD: DEFB 0,0,0,0,0
00

```

```

F565 1B0900 1104      DEFB 1BH,09H,00      ;MOVE CARRIAGE TO COL. XX
F568 1B31 1105      DEFB 1BH,31H      ;SET TAB THERE
F56A 00000000 1106      DEFB 0,0,0,0,0
      00
      1107 ;
      1108 ;
      1109 ;
F56F 0D0A 1110 CRLF:      DEFB 0DH,0AH
F571 00000000 1111      DEFB 0,0,0,0,0,0,0,0
      000000
      1112 ;
      1113 ;
      1114 ;CRTLC:      DEFB 0      ;CRT COLUMN COUNT
F578 0C 1115 LPLC:      DEFB 12      ;PRT COLUMN COUNT
F579 0C 1116 LFMG:      DEFB 12      ;PRT LEFT MARGIN
F57A 00 1117 ESCKEY:      DEFB 0      ;NO ESCAPE KEY SEQUENCE
      1118 ;
      1119 ;
      1120 ;
      1121 ;TAB POSITION TABLE
      1122 ;
      1123 ;
F57B 050A0F14 1124 TABTBL:      DEFB 5,10,15,20,25,30,35,40,45,50
      191E2328
      2D32
F585 373C4146 1125      DEFB 55,60,65,70,75,80,85,90,95,100
      4B50555A
      5F64
F58F 696E7378 1126      DEFB 105,110,115,120,125,130,135,140,0
      7D82878C
      00
      1127 ;
      1128 ;
      1129 ;
      1130 ;
      1131      INCLUDE INTRSV.ASM
      1132 ;*****
      1133 ;*
      1134 ;*      INTERRUPT SERVICE ROUTINES FOR KEYBOARD      *
      1135 ;*      INPUT AND REAL-TIME CLOCK FUNCTIONS      *
      1136 ;*
      1137 ;*
      1138 ;*****
      1139 ;
      1140 ;
      1141 ;
      1142 ;
F598 3A30FF 1143 KBDST: LD      A,(FIFCNT)      ;GET INPUT FIFO BYTECOUNT
F59B B7 1144      OR      A      ;TEST IF EQUAL ZERO
F59C C8 1145      RET      Z      ;EXIT WITH A=0 IF QUEUE IS EMPTY
F59D 3EFF 1146      LD      A,255
F59F C9 1147      RET      ;ELSE SET A=255 TO IND DATA RDY
      1148 ;
      1149 ;
      1150 ;
F5A0 CD98F5 1151 KBDIN: CALL      KBDST
F5A3 28FB 1152      JR      Z,KBDIN-$      ;LOOP UNTIL KEYBOARD INPUT READY
F5A5 E5 1153      PUSH     HL
F5A6 CDBFF5 1154      CALL      REMOVE      ;GET CHARACTER FROM INPUT QUEUE
F5A9 E1 1155      POP      HL
F5AA C9 1156      RET
      1157 ;
      1158 ;
      1159 ;
      1160 ;
      1161 ;
F5AB EE20 1162      XOR      00100000B      ;ELSE TOGGLE BIT 5 OF THE CHAR

```

ROM LISTINGS
 MONITOR ROM VERSION 1.0 (U64 + U63)

```

F5AD 4F      1163 STASH3: LD      C,A
F5AE 2130FF  1164      LD      HL,FIFCNT      ;BUMP INPUT FIFO CHARACTER COUNT
F5B1 7E      1165      LD      A,(HL)
F5B2 3C      1166      INC     A
F5B3 FE10    1167      CP      16
F5B5 D0      1168      RET     NC              ;EXIT NOW IF FIFO IS FULL
F5B6 77      1169      LD      (HL),A          ; ELSE INCREMENT FIFO COUNT
F5B7 2131FF  1170      LD      HL,FIFIN        ;POINT HL TO FIFO INPUT OFFSET
F5BA CDC6F5  1171      CALL    INDEX
F5BD 71      1172      LD      (HL),C          ;STORE CHARACTER IN FIFO @ HL
F5BE C9      1173      RET
                1174 ;
                1175 ;
                1176 ;
                1177 ;
F5BF 2130FF  1178 REMOVE: LD      HL,FIFCNT
F5C2 35      1179      DEC     (HL)
F5C3 2132FF  1180      LD      HL,FIFOUT        ;POINT HL TO FIFO OUTPUT OFFSET
F5C6 7E      1181 INDEX: LD      A,(HL)
F5C7 3C      1182      INC     A
F5C8 E60F    1183      AND     00001111B        ;INCREMENT FIFO POINTER
F5CA 77      1184      LD      (HL),A          ; MODULO 16 AND REPLACE
F5CB 2120FF  1185      LD      HL,FIFO
F5CE 85      1186      ADD     A,L              ;INDEX INTO FIFO BY OFFSET IN A
F5CF 6F      1187      LD      L,A
F5D0 7E      1188      LD      A,(HL)
F5D1 C9      1189      RET
                1190 ;
                1191 ;
                1192 ;      SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
                1193 ;
F5D2 2166FF  1194 DSKTMR: LD      HL,MOTOR      ;DECREMENT DISK TURN-OFF TIMER
F5D5 35      1195      DEC     (HL)
F5D6 C0      1196      RET     NZ              ;EXIT IF NOT TIMED OUT YET
F5D7 DB1C    1197      IN      A,(BITDAT)
F5D9 E6F8    1198      AND     11111000B        ;DISABLE ALL DRIVE SELECTS AND
F5DB D31C    1199      OUT     (BITDAT),A      ; TURN OFF THE SPINDLE MOTORS
F5DD C9      1200      RET
                1201 ;
                1202 ;
                1203 ;
                1204 ;
                1205 ;      -- INTERRUPT SERVICE ROUTINE FOR PARALLEL KEYBOARD --
                1206 ;
F5DE ED7335FF 1207 KEYSRV: LD      (SPSAVE),SP    ;SAVE USER STACK POINTER AND
F5E2 3157FF  1208      LD      SP,TMPSTK+32      ; SWITCH TO LOCAL STACK
F5E5 E5      1209      PUSH    HL
F5E6 D5      1210      PUSH    DE
F5E7 C5      1211      PUSH    BC
F5E8 F5      1212      PUSH    AF              ;SAVE MACHINE STATE
F5E9 DB1E    1213      IN      A,(KBDDAT)        ;READ KEYBOARD INPUT PORT
F5EB 2F      1214      CPL
F5EC E67F    1215      AND     01111111B
F5EE CDADF5  1216      CALL    STASH3
F5F1 F1      1217      POP     AF
F5F2 C1      1218      POP     BC
F5F3 D1      1219      POP     DE
F5F4 E1      1220      POP     HL
F5F5 ED7B35FF 1221      LD      SP,(SPSAVE)
F5F9 FB      1222      EI                      ;RE-ENABLE INTERRUPTS AND RETURN
F5FA ED4D    1223      RETI
                1224 ;
                1225 ;
                1226 ;
                1227 ;      -- INTERRUPT SERVICE ROUTINE FOR ONE SECOND TIMER --
                1228 ;
F5FC ED7335FF 1229 TIMER: LD      (SPSAVE),SP    ;SAVE USER STACK POINTER AND

```

```

F600 3157FF 1230 LD SP,TMPSTK+32 ; SWITCH TO LOCAL STACK
F603 E5 1231 PUSH HL
F604 D5 1232 PUSH DE
F605 C5 1233 PUSH BC
F606 F5 1234 PUSH AF
F607 CDD2F5 1235 CALL DSKTMR ;GO SRVCE THE DSK TURN OFF TIMER
F60A F1 1236 POP AF
F60B C1 1237 POP BC
F60C D1 1238 POP DE
F60D E1 1239 POP HL
F60E ED7B35FF 1240 LD SP,(SPSAVE)
F612 FB 1241 EI ;RE-ENABLE INTERRUPTS AND RETURN
F613 ED4D 1242 RETI
      1243 ;
      1244 ;
      1245 ;
F615 ED7335FF 1246 MILLI: LD (SPSAVE),SP ;SAVE USER STACK POINTER AND
F619 3157FF 1247 LD SP,TMPSTK+32 ; SWITCH TO LOCAL STACK
F61C E5 1248 PUSH HL
F61D F5 1249 PUSH AF
F61E 2A6DFF 1250 LD HL,(INDTMR)
F621 2B 1251 DEC HL ;DECREMENT INDEX PERIOD TIMER
F622 226DFF 1252 LD (INDTMR),HL
F625 F1 1253 POP AF
F626 E1 1254 POP HL
F627 ED7B35FF 1255 LD SP,(SPSAVE)
F62B FB 1256 EI
F62C ED4D 1257 RETI
      1258 ;
      1259 ;
      1260 ;
      1261 ;
      1262 ;
      1263 ; POLLED MODE I/O ROUTINES FOR SIO CHANEL B
      1264 ;
F62E DB07 1265 SIOST: IN A,(SIOCPB) ;GET SIO STATUS REGISTER
F630 E601 1266 AND 00000001B
F632 C8 1267 RET Z ;ACC=0 IF NO DATA AVAILABLE
F633 3EFF 1268 LD A,255
F635 C9 1269 RET
      1270 ;
      1271 ;
F636 CD2EF6 1272 SIOIN: CALL SIOST ;TEST CONSOLE STATUS
F639 28FB 1273 JR Z,SIOIN-$ ;LOOP UNTIL DATA IS
F63B DB05 1274 IN A,(SIODPB) ; READY AT SIO DATA PORT
F63D E67F 1275 AND 01111111B
F63F C9 1276 RET
      1277 ;
      1278 ;
F640 F5 1279 SIOOUT: PUSH AF
F641 DB07 1280 SIOX1: IN A,(SIOCPB)
F643 E604 1281 AND 00000100B ;TEST TBE STATUS BIT
F645 28FA 1282 JR Z,SIOX1-$
F647 F1 1283 POP AF
F648 D305 1284 OUT (SIODPB),A ;OUTPUT DATA TO SIO
F64A C9 1285 RET
      1286 ;
      1287 ;
      1288 ;
      1289 ;
      1290 INCLUDE CRTOUT.ASM
      1291 ;*****
      1292 ;* *
      1293 ;* MEMORY-MAPPED CRT OUTPUT DRIVER *
      1294 ;* *
      1295 ;* *
      1296 ;*****

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

```

1297 ;
1298 ;
0030 1299 CRTBAS EQU CRTMEM.SHR.8 ;STARTING PAGE# OF 3K CRT SPACE
003C 1300 CRTTOP EQU CRTMEM+3072.SHR.8 ;ENDING PAGE# OF CRT SPACE
1301 ;
1302 ;
F64B E5 1303 CRTOUT: PUSH HL
F64C D5 1304 PUSH DE
F64D C5 1305 PUSH BC
F64E CBBF 1306 RES 7,A
F650 4F 1307 LD C,A
F651 F3 1308 DI
F652 ED7335FF 1309 LD (SPSAVE),SP
F656 3157FF 1310 LD SP,TMPSTK+32 ;POINT SP TO TOP OF LOCAL STACK
F659 DB1C 1311 IN A,(BITDAT)
F65B CBFF 1312 SET 7,A ;SELECT ROM/CRT MEMORY BANK
F65D D31C 1313 OUT (BITDAT),A
1314 ;
1315 ; FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN
1316 ;
F65F 2173FF 1317 LD HL,CHRSABV ;GET CHAR NOW OVERLAYED BY CURSOR
F662 46 1318 LD B,(HL)
F663 2A71FF 1319 LD HL,(CURSOR) ;LOAD HL WITH CURSOR POINTER
F666 7C 1320 LD A,H
F667 E60F 1321 AND 00001111B ;A LITTLE INSURANCE THAT HL CAN'T
F669 F630 1322 OR CRTBAS ;EVER POINT OUTSIDE THE CRT MEMORY
F66B 67 1323 LD H,A
F66C 70 1324 LD (HL),B ;REMOVE CURSOR BY RESTORING CHAR
1325 ;
1326 ; PROCESS CHARACTER PASSED IN C
1327 ;
F66D CD90F6 1328 CALL OUTCH
1329 ;
1330 ; NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION
1331 ;
F670 7E 1332 LD A,(HL) ;GET CHAR AT NEW CURSOR LOCATION
F671 3273FF 1333 LD (CHRSABV),A ;SAVE FOR NXT TIME 'CRTOUT' IS CLD
F674 FE20 1334 CP ' ' ;TEST IF CHARACTER IS A SPACE
F676 CBFF 1335 SET 7,A ;THEN TURN ON BIT 7 TO ENBL BLNK
F678 2003 1336 JR NZ,CRT2-$ ;JUMP IF CHARACTER IS NON-BLANK
F67A 3A74FF 1337 LD A,(CSRCHR) ;ELSE GET CHAR USED FOR CURSOR
F67D 77 1338 CRT2: LD (HL),A ;STORE CHAR IN A AS CURSOR MARK
F67E 2271FF 1339 LD (CURSOR),HL ;SAVE HL AS CURSOR POINTER
1340 ;
F681 ED7B35FF 1341 LD SP,(SPSAVE)
F685 DB1C 1342 IN A,(BITDAT)
F687 CBBF 1343 RES 7,A ;SWITCH BACK THE LOWER 16K OF RAM
F689 D31C 1344 OUT (BITDAT),A
F68B FB 1345 EI ;INTERRUPTS ARE SAFE AGAIN
F68C C1 1346 POP BC
F68D D1 1347 POP DE
F68E E1 1348 POP HL
F68F C9 1349 RET
1350 ;
1351 ;
1352 ;
F690 1176FF 1353 OUTCH: LD DE,LEADIN
F693 1A 1354 LD A,(DE) ;GET LEAD-IN SEQUENCE STATE
F694 B7 1355 OR A
F695 C29BF7 1356 JP NZ,MULTI ;JUMP IF IN A LEAD-IN SEQUENCE
F698 79 1357 LD A,C ; ELSE PROCESS CHARACTER IN C
F699 FE20 1358 CP ' '
F69B 380F 1359 JR C,CONTRL-$ ;JUMP IF A CONTROL CHARACTER
F69D 71 1360 DISPLA: LD (HL),C ; ELSE STORE DISPLAYABLE CHARACTER
F69E 23 1361 INC HL ; AND ADVANCE POINTER TO NEXT COL
F69F 7D 1362 LD A,L
F6A0 E67F 1363 AND 01111111B ;EXTRACT COLUMN# FROM HL

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F6A2	FE50	1364	CP	80	
F6A4	D8	1365	RET	C	
F6A5	CD12F7	1366	CALL	RETURN	;EXIT IF NOT PAST COLUMN 79
F6A8	CD6DF7	1367	CALL	LFEED	;ELSE DO AUTOMATIC CARRIAGE RET
F6AB	C9	1368	RET		; AND LINEFEED
		1369 ;			
		1370 ;			
		1371 ;			
F6AC	35	1372	CONTRL: PUSH	HL	
F6AD	21BAF6	1373	LD	HL,CTLTAB	;SEARCH FOR CONTROL CHARACTER
F6B0	010D00	1374	LD	BC,CTLSIZ/3	; HANDLING SUBROUTINE IN TABLE
F6B3	CD56F3	1375	CALL	SEARCH	
F6B6	E1	1376	POP	HL	
F6B7	C0	1377	RET	NZ	;EXIT IF NOT IMPLEMENTED
F6B8	C5	1378	PUSH	BC	
F6B9	C9	1379	RET		;DO SNEAKY JUMP TO PRESERVE REGS
		1380			
F6BA	1F	1381	CTLTAB: DEFB	' '-64	
F6BB	1E	1382	DEFB	' -'-64	
F6BC	1B	1383	DEFB	' ,'-64	
F6BD	1A	1384	DEFB	' Z'-64	
F6BE	18	1385	DEFB	' X'-64	
F6BF	11	1386	DEFB	' Q'-64	
F6C0	0D	1387	DEFB	' M'-64	
F6C1	0C	1388	DEFB	' L'-64	
F6C2	0B	1389	DEFB	' K'-64	
F6C3	0A	1390	DEFB	' J'-64	
F6C4	09	1391	DEFB	' I'-64	
F6C5	08	1392	DEFB	' H'-64	
F6C6	07	1393	DEFB	' G'-64	
		1394			
F6C7	07F7	1395	DEFW	BELL	;CTL-G IS THE BELL
F6C9	E9F6	1396	DEFW	BAKSPC	;CTL-H IS CURSOR LEFT
F6CB	F7F6	1397	DEFW	TAB	;CTL-I IS TAB
F6CD	6DF7	1398	DEFW	LFEED	;CTL-J IS CURSOR DOWN
F6CF	57F7	1399	DEFW	UPCSR	;CTL-K IS CURSOR UP
F6D1	EFF6	1400	DEFW	FORSPC	;CTL-L IS CURSOR RIGHT
F6D3	12F7	1401	DEFW	RETURN	;CTL-M IS CARRIAGE RETURN
F6D5	3CF7	1402	DEFW	CLREOS	;CTL-Q IS CLEAR TO END-OF-SCREEN
F6D7	2EF7	1403	DEFW	CLREOL	;CTL-X IS CLEAR TO END-OF-LINE
F6D9	17F7	1404	DEFW	CLRSCN	;CTL-Z IS CLEAR SCREEN
F6DB	E1F6	1405	DEFW	ESCAPE	;CTL-, IS ESCAPE
F6DD	97F7	1406	DEFW	HOMEUP	;CTL- IS HOME UP
F6DF	E5F6	1407	DEFW	STUFF	;CTL- IS DISPLAY CONTROL CHARS
		1408			
0027		1409	CTLSIZ EQU	\$-CTLTAB	
		1410 ;			
		1411 ;			
F6E1	3E01	1412	ESCAPE: LD	A,1	
F6E3	12	1413	LD	(DE),A	;SET LEAD-IN SEQUENCE STATE
F6E4	C9	1414	RET		; FOR XY CURSOR POSITIONING MODE
		1415 ;			
		1416 ;			
F6E5	3E04	1417	STUFF: LD	A,4	
F6E7	12	1418	LD	(DE),A	;SET LEAD-IN SEQUENCE STATE
F6E8	C9	1419	RET		; FOR CONTROL CHAR OUTPUT MODE
		1420 ;			
		1421 ;			
F6E9	7D	1422	BAKSPC LD	A,L	;CHECK FOR LEFT MARGIN
F6EA	E67F	1423	AND	01111111B	
F6EC	C8	1424	RET	Z	;ABORT IF IN LEFTMOST COLUMN
F6ED	2B	1425	DEC	HL	;BACK UP CURSOR POINTER
F6EE	C9	1426	RET		
		1427 ;			
		1428 ;			
F6EF	7D	1429	FORSPC: LD	A,L	;CHECK FOR RIGHTMOST COLUMN
F6F0	E67F	1430	AND	01111111B	

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F6F2	FE4F	1431	CP	79	
F6F4	D0	1432	RET	NC	;DO NOTHING IF ALREADY THERE
F6F5	23	1433	INC	HL	
F6F6	C9	1434	RET		;ELSE ADVANCE THE CURSOR POINTER
		1435 ;			
		1436 ;			
F6F7	110800	1437	TAB: LD	DE,8	;TABS ARE EVERY 8 COLUMNS
F6FA	7D	1438	LD	A,L	;GET COLUMN COMPONENT OF
F6FB	E678	1439	AND	01111000B	; PREVIOUS TAB POSITION
F6FD	83	1440	ADD	A,E	
F6FE	FE50	1441	CP	80	;EXIT IF NEXT TAB COLUMN WOULD
F700	D0	1442	RET	NC	; BE PAST THE RIGHT MARGIN
F701	7D	1443	LD	A,L	
F702	E6F8	1444	AND	11111000B	;ELSE INCREMENT THE CURSOR
F704	6F	1445	LD	L,A	; POINTER FOR REAL
F705	19	1446	ADD	HL,DE	
F706	C9	1447	RET		
		1448 ;			
		1449 ;			
F707	DB1C	1450	BELL: IN	A,(BITDAT)	
F709	CBEF	1451	SET	5,A	;TOGGLE BIT 5 OF SYSTEM PIO TO
F70B	D31C	1452	OUT	(BITDAT),A	; TRIGGER BELL HARDWARE TO SOUND
F70D	CBAF	1453	RES	5,A	
F70F	D31C	1454	OUT	(BITDAT),A	
F711	C9	1455	RET		
		1456 ;			
		1457 ;			
F712	7D	1458	RETURN: LD	A,L	
F713	E680	1459	AND	10000000B	
F715	6F	1460	LD	L,A	;MOVE CURSOR POINTER BACK
F716	C9	1461	RET		; TO START OF LINE
		1462 ;			
		1463 ;			
F717	210030	1464	CLRSCN: LD	HL,CRTMEM	
F71A	E5	1465	PUSH	HL	
F71B	110130	1466	LD	DE,CRTMEM+1	
F71E	01000C	1467	LD	BC,24*128	
F721	3620	1468	LD	(HL),' '	
F723	EDB0	1469	LDIR		;FILL CRT MEMORY WITH SPACES
F725	E1	1470	POP	HL	;POINT TO HOME CURSOR POSITION
F726	3E17	1471	LD	A,23	
F728	3275FF	1472	LD	(BASE),A	;MAKE BASE LINE# BE 23 AND
F72B	D314	1473	OUT	(SCROLL),A	; STORE IN SCROLL REGISTER
F72D	C9	1474	RET		
		1475 ;			
		1476 ;			
F72E	E5	1477	CLREOL: PUSH	HL	;SAVE CURSOR POINTER
F72F	7D	1478	LD	A,L	
F730	E67F	1479	AND	01111111B	;GET COLUMN# COMPONENT OF
F732	4F	1480	LD	C,A	; CURSOR POINTER INTO C
F733	3E50	1481	LD	A,80	;CALCULATE HOW MANY CHARACTERS
F735	91	1482	SUB	C	; REMAIN ON CURRENT LINE
F736	47	1483	LD	B,A	
F737	CD91F7	1484	CALL	CLR	;CLEAR REST OF LINE @ HL
F73A	E1	1485	POP	HL	
F73B	C9	1486	RET		
		1487 ;			
		1488 ;			
F73C	CD2EF7	1489	CLREOS: CALL	CLREOL	;CLEAR REMAINDER OF CURRENT ROW
F73F	E5	1490	PUSH	HL	
F740	3A75FF	1491	LD	A,(BASE)	
F743	4F	1492	LD	C,A	;COPY BASE SCREEN ROW# TO C
F744	7D	1493	CLRS1: LD	A,L	
F745	17	1494	RLA		
F746	7C	1495	LD	A,H	
F747	17	1496	RLA		;GET ROW# COMPONENT OF HL INTO A
F748	E61F	1497	AND	00011111B	

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F74A	B9	1498	CP	C	;SEE IF HL IS AT BTM ROW OF SCRNB
F74B	2808	1499	JR	Z,CLRS2-\$; AND LEAVE CLEAR LOOP IF SO
F74D	CD62F7	1500	CALL	DNCSR	;ELSE POINT HL TO NEXT ROW DOWN
F750	CD8BF7	1501	CALL	CLRLIN	; AND FILL THAT LINE WITH SPACES
F753	18EF	1502	JR	CLRS1-\$	
		1503			
F755	E1	1504	CLRS2: POP	HL	;RESTORE ORIGINAL CURSOR POINTER
F756	C9	1505	RET		
		1506 ;			
		1507 ;			
F757	1180FF	1508	UPCSR: LD	DE,-128	;SUBTRACT 1 FROM ROW# COMPONENT
F75A	19	1509	ADD	HL,DE	; OF CURSOR POINTER IN HL
F75B	7C	1510	LD	A,H	
F75C	FE30	1511	CP	CRTBAS	;CHECK FOR UNDERFLOW OF POINTER
F75E	D0	1512	RET	NC	
F75F	263B	1513	LD	H,CRTTOP-1	;WRAP CURSOR AROUND MODULO 3K
F761	C9	1514	RET		
		1515 ;			
		1516 ;			
F762	118000	1517	DNCSR: LD	DE,128	;ADD 1 TO ROW# COMPONENT
F765	19	1518	ADD	HL,DE	; OF CURSOR POINTER IN HL
F766	7C	1519	LD	A,H	
F767	FE3C	1520	CP	CRTTOP	;CHECK FOR OVERFLOW OF POINTER
F769	D8	1521	RET	C	
F76A	2630	1522	LD	H,CRTBAS	;RESET POINTER MODULO 128*24
F76C	C9	1523	RET		
		1524 ;			
		1525 ;			
		1526 ;			
F76D	7D	1527	LFEED: LD	A,L	
F76E	17	1528	RLA		
F76F	7C	1529	LD	A,H	
F770	17	1530	RLA		;EXTRACT ROW# COMPONENT OF HL
F771	E61F	1531	AND	00011111B	
F773	4F	1532	LD	C,A	;COPY ROW# INTO C FOR SCROLL TEST
F774	CD62F7	1533	CALL	DNCSR	;MOVE CURSOR TO NEXT ROW DOWN
F777	3A75FF	1534	LD	A,(BASE)	;TEST IF CURSOR WAS ON BOTTOM ROW
F77A	B9	1535	CP	C	;OF SCREEN BEFORE MOVING DOWN
F77B	C0	1536	RET	NZ	;EXIT IF NOT AT BOTTOM
		1537			
F77C	E5	1538	PUSH	HL	;ELSE PREP TO SCROLL SCREEN UP
F77D	CD8BF7	1539	CALL	CLRLIN	;FILL NEW BOTTOM LINE WITH SPACES
F780	29	1540	ADD	HL,HL	
F781	7C	1541	LD	A,H	;GET ROW# COMPONENT OF HL INTO A
F782	E61F	1542	AND	00011111B	
F784	3275FF	1543	LD	(BASE),A	;STORE NEW BASE LINE#
F787	D314	1544	OUT	(SCROLL),A	;NOW SCROLL UP NEW BLNK BTM LINE
F789	E1	1545	POP	HL	
F78A	C9	1546	RET		
		1547 ;			
		1548 ;			
F78B	7D	1549	CLRLIN: LD	A,L	
F78C	E680	1550	AND	10000000B	;POINT HL TO FIRST COLUMN OF ROW
F78E	6F	1551	LD	L,A	
F78F	0650	1552	LD	B,80	
F791	3620	1553	CLR: LD	(HL),' '	;STORE ASCII SPACES AT ADRS IN HL
F793	23	1554	INC	HL	; AND INCREMENT HL
F794	10FB	1555	DJNZ	CLR-\$;REPEAT NUMBER OF TIMES GIVEN BY B
F796	C9	1556	RET		
		1557 ;			
		1558 ;			
F797	0E20	1559	HOMEUP: LD	C,' '	;FAKE-OUT CURSOR ADRSNG ROUTINE
F799	1817	1560	JR	SETROW-\$; TO DO HOMEUP ALMOST FOR FREE
		1561 ;			
		1562 ;			
F79B	EB	1563	MULTI: EX	DE,HL	;UNCONDITIONALLY RESET THE LEAD-IN
F79C	3600	1564	LD	(HL),0	; STATE TO ZERO BEFORE GOING ON


```

F79E EB 1565 EX DE,HL
F79F FE01 1566 CP 1
F7A1 2008 1567 JR NZ,M2TST-$
F7A3 79 1568 SETXY: LD A,C ;GET SECOND CHAR OF SEQUENCE
F7A4 FE3D 1569 CP '='
F7A6 C0 1570 RET NZ ;ABORT SEQUENCE IF NOT '='
F7A7 3E02 1571 LD A,2
F7A9 12 1572 LD (DE),A ;MAKE LEADIN=2 NEXT TIME
F7AA C9 1573 RET
1574
F7AB FE02 1575 M2TST: CP 2
F7AD 2019 1576 JR NZ,M3TST-$
F7AF 3E03 1577 LD A,3
F7B1 12 1578 LD (DE),A ;MAKE LEADIN=3 NEXT TIME
F7B2 3A75FF 1579 SETROW: LD A,(BASE) ;ARRIVE HERE ON THIRD CHARACTER
F7B5 81 1580 ADD A,C ; OF ESC,'=',ROW,COL SEQUENCE
F7B6 D61F 1581 SUB '-'-1
F7B8 D618 1582 SETR2: SUB 24
F7BA 30FC 1583 JR NC,SETR2-$ ;MAKE SURE ROW# IS BTWN 0 AND 23
F7BC C618 1584 ADD A,24
F7BE F660 1585 OR CRTMEM.SHR.7 ;MERGE IN MSB'S OF CRT MEMORY
F7C0 67 1586 LD H,A
F7C1 2E00 1587 LD L,0
F7C3 CB3C 1588 SRL H
F7C5 CB1D 1589 RR L
F7C7 C9 1590 RET
1591
F7C8 FE03 1592 M3TST: CP 3
F7CA 200C 1593 JR NZ,M4TST-$
F7CC 79 1594 SETCOL: LD A,C ;ARRIVE HERE ON FOURTH CHARACTER
F7CD D620 1595 SUB '-' ; OF ESC,'=',ROW,COL SEQUENCE
F7CF D650 1596 SETC2: SUB 80
F7D1 30FC 1597 JR NC,SETC2-$ ;MAKE SURE COL# IS BTWN 0 AND 79
F7D3 C650 1598 ADD A,80
F7D5 B5 1599 OR L ;MERGE IN COL# WITH L
F7D6 6F 1600 LD L,A
F7D7 C9 1601 RET
1602
F7D8 CD9DF6 1603 M4TST: CALL DISPLA ;DISPLAY THE CONTROL CHARACTER
F7DB C9 1604 RET ; PASSED IN C
1605 ;
1606 ;
1607 ;
1608 ;
1609 INCLUDE DISKIO.ASM
1610 ;*****
1611 ;* *
1612 ;* DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE *
1613 ;* FOR WESTERN DIGITAL 1771 DISK CONTROLLER *
1614 ;* *
1615 ;* *
1616 ;*****
1617 ;
1618 ;
1619 ; EQUATES FOR DISK CONTROLLER PORTS AND COMMAND CODES
1620 ;
0010 1621 STSREG EQU WD1771+0 ;STATUS REGISTER
0010 1622 CMDREG EQU WD1771+0 ;COMMAND REGISTER
0011 1623 TRKREG EQU WD1771+1 ;TRACK REGISTER
0012 1624 SECREG EQU WD1771+2 ;SECTOR REGISTER
0013 1625 DATREG EQU WD1771+3 ;DATA REGISTER
1626 ;
0088 1627 RDCMD EQU 10001000B ;READ COMMAND
00A8 1628 WRTCMD EQU 10101000B ;WRITE COMMAND
001C 1629 SKCMD EQU 00011100B ;SEEK COMMAND
00D0 1630 FINCMD EQU 11010000B ;FORCE INTR COMMAND
000C 1631 RSTCMD EQU 00001100B ;RESTORE COMMAND

```

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

0004	1632	HLOAD	EQU	00000100B	;RD/WRT HEAD LOAD ENABLE
	1633				
00C9	1634	RET	EQU	0C9H	;SUBROUTINE RETURN INSTR OPCODE
0066	1635	NMIVEC	EQU	0066H	;THE NON-MASKABLE INTERRUPT IS
	1636				;USED FOR DATA SYNCRONIZATION BTWN
	1637				;THE Z-80 AND 1771 DISK CNTRLR
	1638				
000B	1639	RECNT	EQU	11	;NUMBER OF ERROR RETRY
	1640				
	1641				
F7DC 79	1642	SELECT:	LD	A,C	;GET UNIT# PASSED IN C AND
F7DD 0C	1643		INC	C	;INC DIRVE BY 1
F7DE FE03	1644		CP	3	; CHECK FOR MAXIMUM VALID#
F7E0 D0	1645		RET	NC	;ERROR IF NUMBER 3
F7E1 CDE8F8	1646		CALL	RESTMR	;RESET MTR TIMER & GET PORT DATA
F7E4 47	1647		LD	B,A	;SAVE CURRENT DRIVE SELECT DATA
F7E5 E6F8	1648		AND	11111000B	
F7E7 B1	1649		OR	C	;MERGE IN NEW DRIVE UNIT#
F7E8 CD01F9	1650		CALL	TURNON	;SEE IF NEW DRIVE IS READY
F7EB 2007	1651		JR	NZ,SEL2-\$; AND CONTINUE IF ITS READY
F7ED 78	1652		LD	A,B	;ELSE GET BACK PREV DRIVE SELECT
F7EE D31C	1653		OUT	(BITDAT),A	
F7F0 3E80	1654		LD	A,10000000B	
F7F2 B7	1655		OR	A	;RETURN DRIVE NOT READY INDICATION
F7F3 C9	1656		RET		
	1657				
F7F4 215FFF	1658	SEL2:	LD	HL,UNIT	;POINT HL TO DRIVE SELECT DATA
F7F7 7E	1659		LD	A,(HL)	;LOAD A WITH CURRENT UNIT#
F7F8 71	1660		LD	(HL),C	; AND STORE NEW UNIT# FROM C
F7F9 FEFF	1661		CP	255	;TEST IF NO DRIVE HAS BEEN SELCTD
F7FB 2806	1662		JR	Z,SEL3-\$; YET AND SKIP NEXT SEGMENT IF SO
F7FD 23	1663		INC	HL	;POINT TO HEAD POSITION TABLE
F7FE 85	1664		ADD	A,L	; AND ADD IN NEW UNIT# AS INDEX
F7FF 6F	1665		LD	L,A	
F800 DB11	1666		IN	A,(TRKREG)	;GET CURRENT HEAD POSITION
F802 77	1667		LD	(HL),A	; AND STORE IN TABLE @ HL
F803 2160FF	1668	SEL3:	LD	HL,TRKTAB	
F806 7D	1669		LD	A,L	
F807 81	1670		ADD	A,C	;INDEX INTO TABLE TO GET
F808 6F	1671		LD	L,A	; HEAD POSITION OF NEW DRIVE
F809 7E	1672		LD	A,(HL)	
F80A FEFF	1673		CP	255	;TEST IF NEW DRIVE HAS EVER BEEN
F80C 2804	1674		JR	Z,HOME-\$; SELECTED AND DO A HOME IF NOT
F80E D311	1675		OUT	(TRKREG),A	;OUTPUT THE DRIVE'S CURRENT HEAD
F810 AF	1676		XOR	A	; POSITION TO THE TRACK REGISTER
F811 C9	1677		RET		
	1678				
	1679				
	1680				
F812 CDF3F8	1681	HOME:	CALL	READY	;CLEAR DISK CONTROLLER
F815 C8	1682		RET	Z	;EXIT IF DRIVE NOT READY
F816 AF	1683		XOR	A	
F817 3267FF	1684		LD	(TRACK),A	;SET TRACK# IN MEM TO ZERO
F81A 060C	1685	RESTOR:	LD	B,RSTCMD	;LOAD B WITH A RESTORE COMMAND
F81C CDC8F8	1686		CALL	STEP	;EXECUTE HEAD MOVING OPERATION
F81F EE04	1687		XOR	00000100B	;GET TRUE TRACK 0 STATUS
F821 E69C	1688		AND	10011100B	;MASK TO ERROR BITS
F823 C9	1689		RET		;RETURN 1771 STATUS IN A
	1690				
	1691				
	1692				
F824 CDF3F8	1693	SEEK:	CALL	READY	;CLEAR DISK CONTROLLER
F827 C8	1694		RET	Z	;EXIT IF DRIVE NOT READY
F828 064D	1695		LD	B,77	;SET TRACKS+1 FOR 8 INCH
F82A DB1C	1696		IN	A,(1CH)	;READ HARDWARE PORT FOR DRIVE TYPE
F82C E610	1697		AND	00010000B	;MASK BITS
F82E 2002	1698		JR	NZ,EIGHT-\$;IF 8 IN. DRIVES JUMP

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F830	0628	1699	LD	B,40	;ELSE LOAD TRACK # FOR 5 INCH
F832	79	1700	EIGHT: LD	A,C	;GET TRACK# DATA FROM C
F833	B8	1701	CP	B	; CHECK FOR MAXIMUM VALID#
F834	D0	1702	RET	NC	;FORGET IT IF TRACK# LIMIT
F835	3267FF	1703	LD	(TRACK),A	; STORE TRACK# FOR SEEK
F838	D313	1704	OUT	(DATREG),A	;OUTPUT TRACK # TO 1771
F83A	061C	1705	LD	B,SKCMD	;LOAD B WITH A SEEK COMMAND AND
F83C	CDC8F8	1706	CALL	STEP	; GO SEEK WITH PROPER STEP RATE
F83F	E698	1707	AND	10011000B	;MASK TO READY,SEEK AND CRC ERROR
F841	C8	1708	RET	Z	; BITS AND RETURN IF ALL GOOD
		1709			
F842	CD1AF8	1710	CALL	RESTOR	;ELSE TRY TO RE-CALIBRATE HEAD
F845	C0	1711	RET	NZ	;ERROR IF WE CAN'T FIND TRACK 0
F846	79	1712	LD	A,C	
F847	D313	1713	OUT	(DATREG),A	;OUTPUT TRACK# TO 1771
F849	061C	1714	LD	B,SKCMD	
F84B	CDC8F8	1715	CALL	STEP	;TRY TO SEEK THE TRACK AGAIN
F84E	E698	1716	AND	10011000B	
F850	C9	1717	RET		;RETURN FINAL SEEK STATUS IN A
		1718 ;			
		1719 ;			
		1720 ;			
F851	CDF3F8	1721	WRITE: CALL	READY	;CLEAR THE DISK CONTROLLER
F854	C8	1722	RET	Z	;EXIT IF DRIVE NOT READY
F855	CDE0F8	1723	CALL	FORCE	
F858	CB77	1724	BIT	6,A	
F85A	C0	1725	RET	NZ	;EXIT IF DISK IS WRITE-PROTECTED
F85B	06A8	1726	LD	B,WRTCMD	
F85D	1806	1727	JR	RDWRT-\$	
		1728			
F85F	CDF3F8	1729	READ: CALL	READY	;CLEAR DISK CONTROLLER
F862	C8	1730	RET	Z	;EXIT IF DRIVE NOT READY
F863	0688	1731	LD	B,RDCMD	
F865	226BFF	1732	RDWRT: LD	(IOPTR),HL	;STORE DISK I/O DATA POINTER
F868	2168FF	1733	LD	HL,SECTOR	
F86B	71	1734	LD	(HL),C	;STORE SECTOR# FOR READ/WRITE
F86C	23	1735	INC	HL	
F86D	70	1736	LD	(HL),B	;SAVE READ/WRITE COMMAND BYTE
F86E	23	1737	INC	HL	
F86F	360B	1738	LD	(HL),RECNT	;SET DISK OPERATION RE-TRY COUNT
F871	F3	1739	RW1: DI		;NO INTERRUPTS DURING DISK I/O
F872	216600	1740	LD	HL,NMIVEC	;SAVE BYTE AT NMI VECTOR LOCATION
F875	56	1741	LD	D,(HL)	; IN D FOR DURATION OF READ/WRITE
F876	36C9	1742	LD	(HL),RET	; LOOP AND REPLACE IT WITH A RET
F878	2165FF	1743	LD	HL,RECLEN	
F87B	46	1744	LD	B,(HL)	;B=NUMBER OF BYTES/SECTOR
F87C	0E13	1745	LD	C,DATREG	;C=1771 DATA REGISTER PORT#
F87E	2A6BFF	1746	LD	HL,(IOPTR)	;HL=DISK READ/WRITE DATA POINTER
F881	3A68FF	1747	LD	A,(SECTOR)	;GET SECTOR NUMBER
F884	D312	1748	OUT	(SECREG),A	;OUTPUT SECTOR# TO 1771
F886	CDE0F8	1749	CALL	FORCE	;ISSUE A FORCE INTERRUPT COMMAND
F889	CB6F	1750	BIT	5,A	; TO TEST CURRENT HEAD LOAD STATUS
F88B	3A69FF	1751	LD	A,(CMDTYP)	;GET READ OR WRITE COMMAND BYTE
F88E	2002	1752	JR	NZ,RW2-\$;JUMP IF HEAD IS ALREADY LOADED
F890	F604	1753	OR	HLOAD	; ELSE MERGE IN HLD BIT
F892	CDD8F8	1754	RW2: CALL	CMDOUT	;START THE 1771 DOING IT'S THING
F895	CB6F	1755	BIT	5,A	;TEST IF CMND IS A READ OR WRITE
F897	200D	1756	JR	NZ,WLOOP-\$; AND JUMP TO THE CORRECT LOOP
F899	76	1757	RLOOP: HALT		
F89A	EDA2	1758	INI		
F89C	C299F8	1759	JP	NZ,RLOOP	
F89F	CDD1F8	1760	CALL	BUSY	;LOOP UNTIL 1771 COMES UN-BUSY
F8A2	E69C	1761	AND	10011100B	;MASK OFF TO READY, NOT FOUND, CRC
F8A4	180B	1762	JR	RW3-\$; AND LOST DATA STATUS BITS
		1763			
F8A6	76	1764	WLOOP: HALT		
F8A7	EDA3	1765	OUTI		

ROM LISTINGS
MONITOR ROM VERSION 1.0 (U64 + U63)

F8A9	C2A6F8	1766	JP	NZ,WLOOP	
F8AC	CDD1F8	1767	CALL	BUSY	
F8AF	E6BC	1768	AND	10111100B	;MASK OFF AS ABOVE + WRITE FAULT
F8B1	216600	1769	LD	HL,NMIVEC	
F8B4	72	1770	LD	(HL),D	;RESTORE BYTE @ NMI VECTOR
F8B5	FB	1771	EI		
F8B6	C8	1772	RET	Z	;RETURN IF NO DISK I/O ERRORS
F8B7	216AFF	1773	LD	HL,RETRY	
F8BA	35	1774	DEC	(HL)	;DECREMENT RE-TRY COUNT AND
F8BB	2002	1775	JR	NZ,RW4-\$; EXECUTE COMAND AGAIN IF NOT=0
F8BD	B7	1776	OR	A	
F8BE	C9	1777	RET		;ELSE RETURN 1771 ERROR STATUS
		1778			
F8BF	2167FF	1779	LD	HL,TRACK	
F8C2	4E	1780	LD	C,(HL)	;GET TRACK# FOR CURRENT OPERATION
F8C3	CD24F8	1781	CALL	SEEK	;TRY TO RE-CALIBRATE THE HEAD
F8C6	18A9	1782	JR	RW1-\$; BEFORE READING OR WRITING AGAIN
		1783 ;			
		1784 ;			
		1785 ;			
F8C8	3A64FF	1786	LD	A,(SPEED)	;GET STEP SPEED VARIABLE
F8CB	E603	1787	AND	00000011B	
F8CD	B0	1788	OR	B	;MERGE WITH SEEK/HOME COMMAND IN B
F8CE	CDD8F8	1789	CALL	CMDOUT	;OUTPUT COMMAND AND DELAY
F8D1	DB10	1790	IN	A,(STSREG)	
F8D3	CB47	1791	BIT	0,A	;TEST BUSY BIT FROM
F8D5	20FA	1792	JR	NZ,BUSY-\$; 1771 AND LOOP TILL=0
F8D7	C9	1793	RET		
		1794 ;			
		1795 ;			
		1796 ;			
F8D8	D310	1797	CMDOUT: OUT	(CMDREG),A	;OUTPUT A COMMAND TO THE 1771
F8DA	CDDDF8	1798	CALL	PAUSE	;WAIT 44 MICROSECONDS
F8DD	E3	1799	PAUSE: EX	(SP),HL	
F8DE	E3	1800	EX	(SP),HL	
F8DF	C9	1801	RET		
		1802 ;			
		1803 ;			
		1804 ;			
F8E0	3ED0	1805	LD	A,FINCMD	
F8E2	CDD8F8	1806	CALL	CMDOUT	;ISSUE A FORCE INTERRUPT COMMAND
F8E5	DB10	1807	IN	A,(STSREG)	
F8E7	C9	1808	RET		;RETURN 1771 STATUS REGISTER BITS
		1809 ;			
		1810 ;			
		1811 ;			
F8E8	3EOF	1812	LD	A,15	
F8EA	3266FF	1813	LD	(MOTOR),A	;RE-LOAD MOTOR TURN OFF TIMER
F8ED	CDF2F8	1814	CALL	RES2	
F8F0	DB1C	1815	IN	A,(BITDAT)	;GET STATUS OF SYSTEM PIO
F8F2	C9	1816	RES2: RET		
		1817 ;			
		1818 ;			
		1819 ;			
F8F3	CDE8F8	1820	READY: CALL	RESTMR	;RESET MOTOR TIMER
F8F6	E607	1821	AND	00000111B	;TEST IF MOTORS HAVE BEEN STOPPED
F8F8	C0	1822	RET	NZ	;AND EXIT IF STILL TURNED ON
F8F9	DB1C	1823	IN	A,(BITDAT)	;READ THE SYSTEM PORT
F8FB	E5	1824	PUSH	HL	;SAVE HL
F8FC	215FFF	1825	LD	HL,UNIT	;GET THE DRIVE TO BE SELECTED
F8FF	B6	1826	OR	(HL)	;UPDATE THE A REGISTER
F900	E1	1827	POP	HL	;RESTORE HL
		1828			
		1829 ;			
		1830 ;			
		1831 ;			
		1832 ;			
					TURN ON THE SELECTED DRIVE MOTOR AND START TIMING
					THE ROTATIONAL SPEED TO DETERMINE IF THE DRIVE IS READY

F901	E5	1833	TURNON:	PUSH	HL	;SAVE REGISTERS HL AND BC
F902	C5	1834		PUSH	BC	
F903	D31C	1835		OUT	(BITDAT),A	
F905	3E87	1836		LD	A,10000111B	;PROGRAM CTC1 FOR TIMER MODE
F907	D319	1837		OUT	(CTC1),A	
F909	3E9C	1838		LD	A,156	;INTERRUPT 1000 TIMES/SECOND
F90B	D319	1839		OUT	(CTC1),A	
F90D	21D007	1840		LD	HL,2000	;RESET INDEX PULSE TIMER FOR MAX
F910	226DFF	1841		LD	(INDTMR),HL	; ALLOWABLE SPIN-UP TIME
		1842				
F913	CDE0F8	1843		CALL	FORCE	;GET 1771 STATUS BITS AND MASK TO
F916	E602	1844		AND	00000010B	; INDEX DETECT BIT
F918	47	1845		LD	B,A	;SAVE CURRENT STATE OF BIT IN B
F919	CD53F9	1846	TURN2:	CALL	EDGE	;WAIT FOR THE FIRST CHNG IN INDEX
F91C	3822	1847		JR	C,TURN4-\$;ABORT IF DRIVE NOT READY
F91E	2A6DFF	1848	TURN3:	LD	HL,(INDTMR)	; ELSE GET CURRENT TIMER VALUE
F921	CD53F9	1849		CALL	EDGE	
F924	381A	1850		JR	C,TURN4-\$	
F926	CD53F9	1851		CALL	EDGE	
F929	3815	1852		JR	C,TURN4-\$	
F92B	ED5B6DFF	1853		LD	DE,(INDTMR)	;GET TIMER VALUE AT END OF REVLTN
F92F	ED52	1854		SBC	HL,DE	;CALCULATE PERIOD OF REVOLUTION
F931	226FFF	1855		LD	(PERIOD),HL	
F934	11D200	1856		LD	DE,210	
F937	B7	1857		OR	A	
F938	ED52	1858		SBC	HL,DE	;TEST IF PERIOD IS TOO LONG AND
F93A	30E2	1859		JR	NC,TURN3-\$; TIME ANOTHER REVOLUTION IF TOO
F93C	1E80	1860		LD	E,10000000B	
F93E	1808	1861		JR	TURNX-\$;EXIT WITH DRIVE READY INDICATED
		1862				
F940	DB1C	1863	TURN4:	IN	A,(BITDAT)	;TURN THE MOTOR BACK OFF
F942	E6F8	1864		AND	11111000B	
F944	D31C	1865		OUT	(BITDAT),A	
F946	1E00	1866		LD	E,00000000B	;INDICATE DRIVE-NOT-READY ERROR
F948	3E03	1867	TURNX:	LD	A,00000011B	
F94A	F3	1868		DI		;KILL INTERRUPT FROM CTC CH 2
F94B	D319	1869		OUT	(CTC1),A	
F94D	FB	1870		EI		
F94E	C1	1871		POP	BC	
F94F	E1	1872		POP	HL	;RESTORE HL AND BC
F950	7B	1873		LD	A,E	
F951	B7	1874		OR	A	;RETURN DRIVE READY STATUS IN A
F952	C9	1875		RET		
		1876 ;				
		1877 ;				
		1878 ;				
F953	CDE0F8	1879	EDGE:	CALL	FORCE	;GET CURRENT INDEX DETECT STATE
F956	E602	1880		AND	00000010B	
F958	A8	1881		XOR	B	;COMPARE TO OLD STATE IN B
F959	2009	1882		JR	NZ,EDGE2-\$; AND JUMP IF IT HAS CHANGED
F95B	3A6EFF	1883		LD	A,(INDTMR+1)	
F95E	CB7F	1884		BIT	7,A	;ELSE TEST IF INDEX TIMER HAS
F960	28F1	1885		JR	Z,EDGE-\$;ROLLED OVER & LOOP AGAIN IF NOT
F962	37	1886		SCF		
F963	C9	1887		RET		;RETURN CARRY=1 IF TIMEOUT
		1888				
F964	78	1889	EDGE2:	LD	A,B	
F965	EE02	1890		XOR	00000010B	;COMPLIMENT THE INDEX STATE IN B
F967	47	1891		LD	B,A	
F968	C9	1892		RET		;RETURN WITH CARRY=0
		1893 ;				
		1894 ;				
		1895 ;				
		1896 ;				
		1897 ;				
		1898 ;				
F969	0000	1899	ROMEND:	DEFW	0	;TAIL OF FREE MEMORY LINKED LIST

```

FF00      1900 ;
          1901      ORG      RAM
          1902      INCLUDE MEMORY.ASM
          1903 ;*****
          1904 ;*
          1905 ;*      STORAGE ALLOCATION FOR 256 BYTE SCRATCH RAM      *
          1906 ;*
          1907 ;*****
          1908 ;
          1909 ;
          1910
FF00      1911 VECTAB EQU      $              ;INTERRUPT VECTOR TBL STARTS HERE
FF00      1912 SIOVEC: DEFS    16              ;SPACE FOR 8 VECTORS FOR SIO
FF10      1913 CTCVEC: DEFS    8              ;SPACE FOR 4 VECTORS FOR CTC
FF18      1914 SYSVEC: DEFS    4              ;SPACE FOR 2 VECTORS FOR SYS PIO
FF1C      1915 GENVEC: DEFS    4              ;SPACE FOR 2 VECTORS FOR GEN PIO
          1916 ;
          1917 ;
          1918 ;      KEYBOARD DATA INPUT FIFO VARIABLES
          1919
FF20      1920 FIFO:   DEFS    16              ;CONSOLE INPUT FIFO
FF30      1921 FIFCNT: DEFS    1              ;FIFO DATA COUNTER
FF31      1922 FIFIN:  DEFS    1              ;FIFI INPUT POINTER
FF32      1923 FIFOUT: DEFS    1              ;FIFO OUTPUT POINTER
FF33      1924 LOCK:   DEFS    2              ;SHIFT LOCK CHARACTER+FLAG BYTE
          1925 ;
          1926 ;
          1927 ;      STACK POINTER SAVE AND LOCAL STACK FOR INTERRUPT ROUTINES
          1928
FF35      1929 SPSAVE: DEFS    2              ;USER STACK POINTER SAVE AREA
FF37      1930 TMPSTK: DEFS    32             ;LOCAL STACK FOR INTERRUPTS
          1931 ;
          1932 ;
          1933 ;      CLOCK-TIMER INTERRUPT VARIABLES
          1934
FF57      1935 TIKCNT: DEFS    2              ;BINARY CLOCK TICK COUNTER
FF59      1936 DAY:    DEFS    1              ;CALENDAR DAY
FF5A      1937 MONTH:  DEFS    1              ;      MONTH
FF5B      1938 YEAR:   DEFS    1              ;      YEAR
FF5C      1939 HRS:    DEFS    1              ;CLOCK HOURS REGISTER
FF5D      1940 MINS:   DEFS    1              ;      MINUTES RETISTER
FF5E      1941 SECS:   DEFS    1              ;      SECONDS REGISTER
          1942 ;
          1943 ;
          1944 ;      DISK I/O DRIVER VARIABLES
          1945
FF5F      1946 UNIT:   DEFS    1              ;CURRENTLY SELECTED DISK#
FF60      1947 TRKTAB: DEFS    4              ;4 DRIVE HEAD POSITION TABLE
FF64      1948 SPEED:  DEFS    1              ;SEEK SPEED FOR 1771 COMMANDS
FF65      1949 RECLN:  DEFS    1              ;SECTOR RECORD LENGTH VARIABLE
FF66      1950 MOTOR:  DEFS    1              ;DRIVE MOTOR TURN-OFF TIMER
FF67      1951 TRACK:  DEFS    1
FF68      1952 SECTOR: DEFS    1
FF69      1953 CMDTYP: DEFS    1              ;COMMAND BYTE FOR READS/WRITES
FF6A      1954 RETRY:  DEFS    1              ;DISK OPERATION RE-TRY COUNT
FF6B      1955 IOPTR:  DEFS    2              ;DISK I/O BUFFER POINTER
FF6D      1956 INDTMR: DEFS    2              ;INDEX HOLE CYCLE PERIOD
FF6F      1957 PERIOD: DEFS    2              ;PERIOD OF REVOLUTION OF DISK
          1958 ;
          1959 ;
          1960 ;
          1961 ;      CRT OUTPUT DRIVER VARIABLES
          1962
FF71      1963 CURSOR: DEFS    2              ;CURSOR POINTER
FF73      1964 CHRSAV: DEFS    1              ;CHARACTER OVERLAYED BY CURSOR
FF74      1965 CSRCHR: DEFS    1              ;CHARACTER USED FOR A CURSOR
FF75      1966 BASE:   DEFS    1              ;CURRENT CONTENTS OF SCROLL REG

```

ROM LISTINGS
 MONITOR ROM VERSION 1.0 (U64 + U63)

```

FF76      1967 LEADIN: DEFS      1                ;STATE OF LEAD-IN SEQUENCE HANDLER
          1968 ;
          1969 ;
          1970 ;
          1971 ;          LISTHEAD POINTER FOR DYNAMIC MEMORY ALLOCATION SCHEME
          1972
FF77      1973 FREPTR: DEFS      2
          1974 ;
          1975 ;
          1976 ;          CONSOLE MONITOR PROGRAM VARIABLES
          1977
FF79      1978 PARAM1: DEFS      2                ;STORAGE FOR NUMBERS READ
FF7B      1979 PARAM2: DEFS      2                ; FROM LINE INPUT BUFFER
FF7D      1980 PARAM3: DEFS      2                ; BY 'PARAMS' SUBROUTINE
FF7F      1981 PARAM4: DEFS      2
FF81      1982 ESCFLG: DEFS      1                ;CONSOLE ESCAPE FLAG
FF82      1983 LAST:   DEFS      2                ;LAST ADDRESS USED BY 'MEMDMP'
FF84      1984 LINBUF: DEFS     64                ;CONSOLE LINE INPUT BUFFER
          1985 ;
          1986 ;
          1987
          1988 ;
          1989      END

```

MONITOR ROM VERSION 2.0 (U64)

```

0001 ;*****
0002 ;*
0003 ;*          XEROX 820      MONITOR  ROM
0004 ;*
0005 ;*          VERSION      2.0
0006 ;*
0007 ;*****
0008 ;
0009 ;
0010      PSECT  ABS
EFF0    0011 ROM  EQU  OEFF0H          ;START OF 4K ROM-TRANSFER CODE
F7F0    0012 ROM2SP EQU  OF7F0H        ;START OF ROM 2 SPRING BOARD
0013 ;
0014 ;EQUATES FOR ROUTINE CALL TO ROM 2
0015 ;
F7F0    0016 MEMDMP EQU  ROM2SP        ;MEMORY DUMP ROUTINE
F7F3    0017 BLOCK EQU  MEMDMP+3      ;BLOCK MOVE ROUTINE
F7F6    0018 VIEW  EQU  BLOCK+3       ;MEMORY DISPLAY AND VERIFY
F7F9    0019 FILL  EQU  VIEW+3        ;MEMORY FILL ROUTINE
F7FC    0020 TEST  EQU  FILL+3        ;MEMORY DIAGNOSTICS
F7FF    0021 GOTO  EQU  TEST+3        ;EXECUTION ROUTINE
F802    0022 VERCMD EQU  GOTO+3       ;MEMORY BLOCK COMPARE
F805    0023 TYPE  EQU  VERCMD+3      ;TYPEWRITER MODE
0024 ;
0025 ;
0026 ;
FF00    0027 RAM   EQU  OFF00H        ;START OF 256 BYTE RAM
3000    0028 CRTMEM EQU  3000H        ;BASE OF 4K CRT MEMORY
0029 ;
0030 ;
EFF0    0031      ORG  ROM
0032 ;
0033 ;
0034 ;      COPY ROM CODE TO HIGH MEMORY
0035 ;      ON POWER-UP
0036 ;
EFF0 F3  0037      DI              ;KEEP OTHERS AWAY
EFF1 211000 0038      LD  HL,0010H    ;SET START ADDRESS
EFF4 1100F0 0039      LD  DE,0F000H    ;SET DESTINATION ADDRESS
EFF7 010010 0040      LD  BC,1000H     ;SET LENGTH OF MOVE
EFAA EDB0 0041      LDIR             ;MOVE IT ALL
EFFC C300F0 0042      JP   0F000H      ;JUMP TO THE ROM CODE IN HI MEM
FFFF 00 0043      NOP              ;JUST TO LINE UP BOUNDS
0044 ;
0045 ;
0046      INCLUDE INIT.ASM
0047 ;*****
0048 ;*
0049 ;*      COLD START INITIALIZATION ROUTINE FOR
0050 ;*      CONFIGURING THE SYSTEM AFTER A POWER-ON
0051 ;*      OR PUSHBUTTON RESET.
0052 ;*      XEROX 820 VER. 2.0      28-JULY-1981
0053 ;*
0054 ;*****
0055 ;
0056 ;
0057 ;      -- MONITOR ENTRY POINT TABLE --
0058 ;
F000 C345F0 0059 COLD:  JP   INIT          ;MONITOR COLD ENTRY POINT
F003 C316F1 0060 WARM:  JP   PROMPT        ;MONITOR WARM ENTRY POINT
F006 C368F3 0061 CONST: JP   KBDST         ;CONSOLE STATUS VECTOR
F009 C370F3 0062 CONIN: JP   KBDIN         ;CONSOLE INPUT VECTOR
F00C C321F4 0063 CONOUT: JP  CRTOUT        ;CONSOLE OUTPUT VECTOR
F00F C321F4 0064      JP   CRTOUT        ;CRT OUTPUT VECTOR

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)


```

F012 C3FEF3 0065 JP SIOST ;SIO CHANEL B STATUS VECTOR
F015 C306F4 0066 JP SIOIN ;SIO CHANEL B INPUT VECTOR
F018 C310F4 0067 JP SIOOUT ;SIO CHANEL B OUTPUT VECTOR
F01B C3B0F5 0068 JP SELECT ;DISK DRIVE SELECT
F01E C3ECF5 0069 JP HOME ;HOME R/W HEAD
F021 C3FEF5 0070 JP SEEK ;SEEK TO TRACK
F024 C339F6 0071 JP READ ;READ SECTOR
F027 C32BF6 0072 JP WRITE ;WRITE SECTOR
F02A C314F2 0073 JP DUMP ;DUMP MEMORY CONTENTS
F02D C301F3 0074 JP PUT4HS ;PRINT ADDRESS IN HEX
F030 C306F3 0075 JP PUT2HS ;PRINT DATA IN HEX
F033 C338F3 0076 JP SPACE ;PRINT A SPACE
F036 C34CF3 0077 JP OUTPUT ;PRINT ASCII CHARACTER IN A
F039 C332F3 0078 JP CRLFS ;PRINT CRLF
F03C C33EF3 0079 JP ECHO ;PRINT INPUT CHAR TO CONSOLE
F03F C3F1F2 0080 JP ASCHEX ;CONVERT ASCII TO HEX
F042 C322F3 0081 JP PNEXT ;DISPLAY MESSAGE
0082 ;
0083 ;
0084 ;
0085 ; DO A SHORT POST-RESET DELAY BY FILLING THE
0086 ; 256 BYTE SCRATCH MEMORY WITH ZEROS
0087 ;
F045 F3 0088 INIT: DI
F046 21EDFF 0089 LD HL, RAM+255-2-16 ;POINT TO END OF MONITOR RAM
0090 ;
0091 ;CRC FOR THE FIRST ROM IS IN F7EE AND F7EF
0092 ;CRC FOR THE SECOND ROM IS IN FFEE AND FFEF
0093 ;
F049 3600 0094 INIT1: LD (HL),0 ;FILL 256 BYTE SPACE WITH ZEROS
F04B F9 0095 LD SP,HL ;DO SOMETHING USEFUL TO ADD DELAY
F04C 2D 0096 DEC L ;GO BACKWARD IN ADDRESS (VER. 2.0)
F04D 20FA 0097 JR NZ, INIT1-$ ;LOOP TAKES ABOUT 4 MILLISECONDS
0098 ;
0099 ; STORE ANY NON-ZERO VALUES FOR VARIABLES IN MEMORY
0100 ;
F04F 21C7F0 0101 LD HL, INTAB ;POINT TO DEFAULT VARIABLE TABLE
F052 0600 0102 IN: I2: LD B,0
F054 4E 0103 LD C,(HL) ;BC=DATA BLOCK BYTECOUNT
F055 23 0104 INC HL
F056 5E 0105 LD E,(HL) ;DE=DESTINATION FOR DATA
F057 23 0106 INC HL
F058 56 0107 LD D,(HL)
F059 23 0108 INC HL
F05A EDB0 0109 LDIR ;COPY DATA @ HL TO VARIABLES @ DE
F05C CB7E 0110 BIT 7,(HL)
F05E 28F2 0111 JR Z, INIT2-$ ;LOOP AGAIN IF NOT AT END OF TBL
0112 ;
0113 ; INITIALIZE THE PROGRAMMABLE I/O DEVICES
0114 ;
F060 23 0115 INC HL ;POINT TO I/O INIT DATA TABLE
F061 46 0116 INIT3: LD B,(HL) ;B=INIT LOOP BYTECOUNT
F062 23 0117 INC HL
F063 4E 0118 LD C,(HL) ;C=DEVICE CONTROL PORT#
F064 23 0119 INC HL
F065 EDB3 0120 OTIR ;SEND DATA @ HL TO PORT @ C
F067 CB7E 0121 BIT 7,(HL) ;TEST FOR TABLE END MARKER
F069 28F6 0122 JR Z, INIT3-$ ;LOOP AGAIN IF NOT AT END
0123 ;
0124 ; INITIALIZE THE Z-80 FOR INTERRUPT MODE #2
0125 ;
F06B 3EFF 0126 LD A, VECTAB.SHR.8
F06D ED47 0127 LD I,A ;LOAD I REG WITH MSB OF VECTOR TBL
F06F ED5E 0128 IM 2 ; AND SELECT INTERRUPT MODE 2
0129 ;
0130 ; SELECT STEP SPEED FOR 8" DISC DRIVE AND 5" DISC DRIVE
0131 ; VERSION 2.0

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

0132 ;
F071 DB1C 0133 IN A,(SYSPIO) ;GET DRIVE STATUS
F073 CB67 0134 BIT 4,A ;TEST DRIVE BIT
F075 2805 0135 JR Z,SIGNON-$ ;5" DRIVE USE 20MS STEP RATE
F077 3E02 0136 LD A,02H ;8" DRIVE USE 8MS STEP RATE
F079 3267FF 0137 LD (SPEED),A
0138 ;
0139 ;
0140 ; PRINT SIGNON MESSAGE
0141 ;
F07C FB 0142 SIGNON: EI
F07D CD22F3 0143 CALL PNEXT
F080 1A 0144 DEFB 'Z'-64
F081 2E2E2E58 0145 DEFM '...XEROX 820 VER. 2.0...'
45524F58
20383230
20205645
522E2032
2E302E2E
2E
F09A 0D0A 0146 DEFB CR,LF
F09C 20202041 0147 DEFM ' A - BOOT SYSTEM'
202D2042
4F4F5420
53595354
454D
FOAE 0D0A 0148 DEFB CR,LF
FOBO 20202054 0149 DEFM ' T - TYPEWRITER'
202D2054
59504557
52495445
52
FOC1 0D0A 0150 DEFB CR,LF
FOC3 04 0151 DEFB EOT
FOC4 C303F0 0152 JP WARM ;GO ENTER MONITOR
0153 ;
0154 ;
0155 ;
0156 ;
FOC7 0157 INTAB EQU $ ;INITIALIZATION DATA TABLES
0158 ;
0159 ; INITIALIZE THE Z-80 'I' REGISTER INTERRUPT VECTOR TABLE
0160 ;
FOC7 02 0161 DEFB 2
FOC8 1AFF 0162 DEFW SYSVEC+2
FOCA AEF3 0163 DEFW KEYSRV ;PARALLEL KEYBOARD INTRPT VECTOR
0164
FOCC 02 0165 DEFB 2
FOCD 12FF 0166 DEFW CTCVEC+2
FOCF E5F3 0167 DEFW MILLI ;ONE MILLISECOND INTERRUPT TIMER
0168
FOD1 02 0169 DEFB 2
FOD2 16FF 0170 DEFW CTCVEC+6
FOD4 CCF3 0171 DEFW TIMER ;ONE SECOND TIMER INTRPT VECTOR
0172 ;
0173 ; INITIALIZE DISK I/O DRIVER VARIABLES
0174 ;
FOD6 0B 0175 DEFB 11
FOD7 5FFF 0176 DEFW UNIT
FOD9 FF 0177 DEFB 255 ;FLAG ALL DRIVES AS DE-SELECTED
FODA FFFFFF 0178 DEFB 255,255,255
FODD FFFFFFFF 0179 DEFB 255,255,255,255 ;CLEAR HEAD POSITION TABLE
FOE1 03 0180 DEFB 00000011B ;SELECT SLOWEST SEEK SPEED
FOE2 80 0181 DEFB 128 ;SELECT 128 BYTE SECTOR LENGTH
FOE3 0F 0182 DEFB 15 ;SET MOTOR TURN-OFF TIMER
0183 ;
0184 ; INITIALIZE THE CRT DISPLAY CURSOR

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

		0185 ;			
FOE4	01	0186	DEFB	1	
FOE5	77FF	0187	DEFW	CSRCHR	
FOE7	02	0188	DEFB	02	;USE NON-BLINKING BOX
		0189 ;			
		0190 ;			SET FREE MEMORY POINTER
		0191 ;			
FOE8	02	0192	DEFB	2	
FOE9	7AFF	0193	DEFW	FREPTR	
FOEB	0001	0194	DEFW	100H	;POINT TO FIRST LOCATN AFTER MONITR
		0195 ;			
		0196 ;			
FOED	FF	0197	DEFB	-1	;END OF VARIABLE INIT TABLE
		0198 ;			
		0199 ;			
		0200 ;			
0000		0201 BAUDA	EQU	00H	;CHANEL A BAUD RATE GENETATOR
0004		0202 SIO	EQU	04H	;DUAL SERIAL I/O
0008		0203 GENPIO	EQU	08H	;GENERAL PURPOSE PARALLEL I/O
000C		0204 BAUDB	EQU	0CH	;CHANEL B BAUD RATE GENERATOR
0010		0205 WD1771	EQU	10H	;WESTERN DIGITAL DISK CONTROLLER
0014		0206 SCROLL	EQU	14H	;CRT SCROLL MEMORY SCROLL REG
0018		0207 CTC	EQU	18H	;QUAD COUNTER/TIMER CIRCUIT
001C		0208 SYSPIO	EQU	1CH	;SYSTEM PARALLEL I/O
		0209 ;			
		0210 ;			INITIALIZE SYSTEM PIO FOR USE AS BANK-SWITCH,
		0211 ;			DISK DRIVE SELECT AND PARALLEL KEYBOARD INPUT
		0212 ;			
001C		0213 BITDAT	EQU	SYSPIO+0	
001D		0214 BITCTL	EQU	SYSPIO+1	
001E		0215 KBDDAT	EQU	SYSPIO+2	
001F		0216 KBDCTL	EQU	SYSPIO+3	
		0217			
FOEE	031D	0218	DEFB	3,BITCTL	
FOF0	CF	0219	DEFB	11001111B	;PUT SYSTEM PIO IN BIT MODE
FOF1	38	0220	DEFB	00111000B	;MAKE BITS 5 AND 4 & 3 BE INPUTS
FOF2	40	0221	DEFB	01000000B	;DISABLE INTERRUPTS
		0222 ;			
FOF3	011C	0223	DEFB	1,BITDAT	
FOF5	00	0224	DEFB	00000000B	;DE-SELECT ROMS, ENABLE DRIVE 0
		0225 ;			
FOF6	031F	0226	DEFB	3,KBDCTL	
FOF8	4F	0227	DEFB	01001111B	;PUT KEYBOARD PORT IN INPUT MODE
FOF9	1A	0228	DEFB	SYSEVC+2	;LOAD KEYBOARD INTERRUPT VECTOR
FOFA	83	0229	DEFB	10000011B	;ENABLE INTERRUPTS
		0230 ;			
		0231 ;			
		0232 ;			INITIALIZE CHANELS 2 AND 3 OF THE CTC
		0233 ;			TO GENERATE ONE SECOND INTERRUPTS FROM CTC3
		0234 ;			
0018		0235 CTC0	EQU	CTC+0	;CTC CHANEL 0 PORT#
0019		0236 CTC1	EQU	CTC+1	;CTC CHANEL 1
001A		0237 CTC2	EQU	CTC+2	;CTC CHANEL 2
001B		0238 CTC3	EQU	CTC+3	;CTC CHANEL 3
		0239			
FOFB	0118	0240	DEFB	1,CTC0	
FOFD	10	0241	DEFB	CTCVEC	;BASE INTERRUPT VECTOR FOR CTC
		0242 ;			
FOFE	021A	0243	DEFB	2,CTC2	
F100	27	0244	DEFB	00100111B	;PUT CTC2 IN TIMER MODE
F101	69	0245	DEFB	105	;CTC2 PERIOD=105*256*400 NANOSCNDs
		0246 ;			
F102	021B	0247	DEFB	2,CTC3	
F104	C7	0248	DEFB	11000111B	;PUT CTC3 IN COUNTER MODE
F105	5D	0249	DEFB	93	;CTC3 PERIOD=999936 MICROSECONDS
		0250 ;			
		0251 ;			

```

0252 ;      INITIALIZE SIO CHANEL B FOR ASYNCHRONOUS SERIAL
0253 ;      INTERFACE TO PRINTER OR TERMINAL
0254 ;
0004      0255 SIODPA EQU      SIO+0      ;SIO DATA PORT A
0005      0256 SIODPB EQU      SIO+1      ;SIO DATA PORT B
0006      0257 SIOCPA EQU      SIO+2      ;SIO CONTROL/STATUS PORT A
0007      0258 SIOCPB EQU      SIO+3      ;SIO CONTROL/STATUS PORT B
0259
F106 0A07      0260      DEFB      10,SIOCPB
F108 04      0261      DEFB      4      ;SELECT REGISTER #4
F109 45      0262      DEFB      01000101B ;16X CLOCK, 1 STOP BIT
F10A 01      0263      DEFB      1      ;SELECT REGISTER #1
F10B 04      0264      DEFB      00000100B ;STATUS AFFECTS VECTOR
F10C 03      0265      DEFB      3      ;SELECT REGISTER #3
F10D 41      0266      DEFB      01000001B ;7 BITS/RX CHARACTERS
F10E 05      0267      DEFB      5      ;SELECT REGISTER #5
F10F 2A      0268      DEFB      00101010B ;7 BITS/TX CHARACTER
F110 02      0269      DEFB      2      ;SELECT REGISTER #2
F111 00      0270      DEFB      SIOVEC      ;BASE SIO INTERRUPT VECTOR
0271
F112 010C      0272      DEFB      1,BAUDB
F114 05      0273      DEFB      0101B      ;DEFAULT BAUD RATE=300
0274
F115 FF      0275      DEFB      -1      ;END-OF-TABLE
0276 ;
0277 ;
0278 ;
0279 ;
0280      INCLUDE MON1.ASM
0281 ;*****
0282 ;*
0283 ;*      BASIC HEX MONITOR FOR Z-80 PROCESSORS
0284 ;*
0285 ;*****
0286 ;
0287 ;
0288 ;
0289 ;
F116 CD22F3      0290 PROMPT: CALL      PNEXT
F119 0D0A      0291      DEFB      CR,LF
F11B 2A20      0292      DEFM      '* '
F11D 04      0293      DEFB      EOT
F11E 2187FF      0294      LD      HL,LINBUF
F121 0E50      0295      LD      C,80      ;BUFFER OF 80 CHARS (VER. 2.0)
F123 CD6FF2      0296      CALL     GETLIN      ;INPUT A BUFERED CONSOLE LINE
F126 3835      0297      JR      C,WHAT-$      ;PRINT 'WHAT ?' IF INPUT ERROR
0298
F128 AF      0299      XOR      A
F129 3284FF      0300      LD      (ESCFLG),A
F12C CD32F3      0301      CALL     CRLFS
F12F 3A87FF      0302      LD      A,(LINBUF)      ;GET FIRST CHARACTER IN LINE
F132 FE0D      0303      CP      CR
F134 28E0      0304      JR      Z,PROMPT-$      ;JUMP IF A NULL LINE
F136 216CF1      0305      LD      HL,CMDTAB      ;SEARCH FOR A MATCHING CHARACTER
F139 010D00      0306      LD      BC,CMDsiz/3      ; IN COMMAND SEARCH TABLE
F13C CD94F2      0307      CALL     SEARCH
F13F 201C      0308      JR      NZ,WHAT-$      ;TRY AGAIN IF SEACRH FAILS
F141 C5      0309      PUSH     BC
F142 FD2188FF      0310      LD      IY,LINBUF+1
F146 CD9EF2      0311      CALL     PARAMS      ;INPUT NUMERIC PARAMETERS FROM
F149 DDE1      0312      POP      IX      ; LINE BUFFER AND TEST IF ERROR
F14B 3810      0313      JR      C,WHAT-$
F14D 2A7CFF      0314      LD      HL,(PARAM1)
F150 ED5B7EFF      0315      LD      DE,(PARAM2)
F154 ED4B80FF      0316      LD      BC,(PARAM3)
F158 CD6AF1      0317      CALL     CALLX      ;CALL SUBROUTINE @ IX
F15B 30B9      0318      JR      NC,PROMPT-$      ;GO BACK TO PROMPT IF NO ERRORS

```

ROM LISTINGS
 MONITOR ROM VERSION 2.0 (U64)

```

0319
F15D CD22F3 0320 WHAT: CALL PNEXT
F160 20776861 0321 DEFM ' what ?'
74203F
0322 ; DEFB 'G'-64 ;SAY 'what ?' AND BEEP THE BELL
F167 04 0323 DEFB EOT
F168 18AC 0324 JR PROMPT-$
0325 ;
0326 ;
F16A DDE9 0327 CALLX: JP (IX) ;CALL SUBROUTINE @ IX
0328 ;
0329 ;
0330 ;
F16C 54 0331 CMDTAB: DEFB 'T'
F16D 56 0332 DEFB 'V'
F16E 52 0333 DEFB 'R'
F16F 4F 0334 DEFB 'O'
F170 49 0335 DEFB 'I'
F171 47 0336 DEFB 'G'
F172 58 0337 DEFB 'X'
F173 46 0338 DEFB 'F'
F174 4D 0339 DEFB 'M'
F175 43 0340 DEFB 'C'
F176 42 0341 DEFB 'B'
F177 44 0342 DEFB 'D'
F178 41 0343 DEFB 'A'
F179 93F1 0344 DEFW BOOT ;BOOT FROM DRIVE B
F17B F0F7 0345 DEFW MEMDMP ;DUMP MEMORY IN HEX/ASCII
F17D ADF1 0346 DEFW BOOTALT ;BOOT UP CP/M
F17F F3F7 0347 DEFW BLOCK ;MEMORY BLOCK MOVE
F181 F6F7 0348 DEFW VIEW ;MEMORY EXAMINE/CHANGE
F183 F9F7 0349 DEFW FILL ;FILL MEMORY
F185 FCF7 0350 DEFW TEST ;RAM DIAGNOSTIC
F187 FFF7 0351 DEFW GOTO ;JUMP TO MEMORY LOCATION
F189 44F2 0352 DEFW INCMD ;READ FROM INPUT PORT
F18B 66F2 0353 DEFW OUTCMD ;WRITE TO OUTPUT PORT
F18D B1F1 0354 DEFW DSKCMD ;DISPLAY DISK SECTOR DATA
F18F 02F8 0355 DEFW VERCMD ;MEMORY BLOCK COMPARE
F191 05F8 0356 DEFW TYPE ;TYPEWRITER MODE
0357 ;
0358 ;
0027 0359 CMDSIZ EQU $-CMDTAB
0360 ;
0361 ;
0362 ;*****
0363 ;*
0364 ;* MONITOR COMMAND ACTION ROUTINES PACKAGE *
0365 ;*
0366 ;*****
0367 ;
0368 ;
0369 ;
0370 ;
0371 ;
0372 ; -- DISK BOOT LOADER COMMAND --
0373 ;
F193 0E00 0374 BOOT: LD C,0 ;SELECT DRIVE 0 FOR BOOT LOAD
F195 CDB0F5 0375 BOOT1: CALL SELECT
F198 2043 0376 JR NZ,DSKERR-$
F19A CDEC5 0377 CALL HOME ;HOME HEAD TO TRACK 0
F19D 203E 0378 JR NZ,DSKERR-$ ;ERROR IF NOT READY OR AT TRO
F19F 218000 0379 LD HL,128 ;POINT TO CP/M READ BUFFER
F1A2 0E01 0380 LD C,1 ;SELECT SECTOR 1
F1A4 CD39F6 0381 CALL READ ;READ TRACK 0/ SECTOR 1
F1A7 2034 0382 JR NZ,DSKERR-$
F1A9 F1 0383 POP AF ;CLEAN UP STACK
F1AA C38000 0384 JP 128 ;GO EXECUTE LOADER AT 128

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

0385 ;
0386 ;
0387 ;           ALTERNATE BOOT FROM DRIVE 'B'
0388 ;
F1AD OE01 0389 BOOTALT: LD      C,1           ;LOAD THE DRIVE NUMBER
F1AF 18E4 0390 JR          BOOT1-$         ;CONT WITH NORMAL BOOT ROUTINE
0391 ;
0392 ;
0393 ;           -- DISK SECTOR READ COMMAND --
0394 ;
F1B1 FE03 0395 DSKCMD: CP      3           ;CHECK PARAMETER COUNT
F1B3 37   0396 SCF
F1B4 C0   0397 RET      NZ
F1B5 4D   0398 LD      C,L           ;USE FIRST ARG AS UNIT#
F1B6 CDB0F5 0399 CALL    SELECT
F1B9 2022 0400 JR      NZ,DSKERR-$
F1BB 217EFF 0401 LD      HL,PARAM2
F1BE 4E   0402 LD      C,(HL)       ;USE SECOND ARG AS TRACK#
F1BF CDFEF5 0403 CALL    SEEK
F1C2 2019 0404 JR      NZ,DSKERR-$
F1C4 2180FF 0405 LD      HL,PARAM3
F1C7 4E   0406 LD      C,(HL)       ;USE THIRD ARG AS SECTOR#
F1C8 218000 0407 DSK2: LD      HL,128
F1CB CD39F6 0408 CALL    READ
F1CE CBC7  0409 SET      0,A         ;MARK ERROR BYTE AS DUE TO READ
F1D0 200B  0410 JR      NZ,DSKERR-$
F1D2 218000 0411 LD      HL,128
F1D5 110800 0412 LD      DE,8
F1D8 CD14F2 0413 CALL    DUMP       ;DUMP DISK READ BUFFER AND
F1DB 1814  0414 JR      DSKADR-$   ; PRINT UNIT/TRACK/SECTOR
0415
F1DD F5    0416 DSKERR: PUSH    AF       ;SAVE 1771 STATUS
F1DE CD22F3 0417 CALL    PNEXT
F1E1 6469736B 0418 DEFM    'disk error '
20657272
6F7220
F1EC 04    0419 DEFB    EOT
F1ED F1    0420 POP      AF
F1EE CD06F3 0421 CALL    PUT2HS
F1F1 3E55  0422 DSKADR: LD      A,'U'   ;PRINT ERROR STATUS IN HEX
F1F3 CD4CF3 0423 CALL    OUTPUT        ;NOW DISPLAY UNIT/TRACK/SECTOR
F1F6 3A5FFF 0424 LD      A,(UNIT)
F1F9 CD06F3 0425 CALL    PUT2HS       ;PRINT DRIVE UNIT#
F1FC 3E54  0426 LD      A,'T'
F1FE CD4CF3 0427 CALL    OUTPUT
F201 3A6AFF 0428 LD      A,(TRACK)
F204 CD06F3 0429 CALL    PUT2HS       ;PRINT TRACK# IN HEX
F207 3E53  0430 LD      A,'S'
F209 CD4CF3 0431 CALL    OUTPUT
F20C 3A6BFF 0432 LD      A,(SECTOR)
F20F CD06F3 0433 CALL    PUT2HS       ;PRINT SECTOR# IN HEX
F212 B7    0434 OR      A
F213 C9    0435 RET
0436 ;
0437 ;
0438 ;
0439 ;
F214 E5    0440 DUMP:  PUSH    HL       ;SAVE STARTING ADDRESS
F215 CD01F3 0441 CALL    PUT4HS      ;PRINT STARTING ADDRESS IN HEX
F218 CD38F3 0442 CALL    SPACE
F21B 0610  0443 LD      B,16
F21D 7E    0444 DUMP2: LD      A,(HL)   ;GET A DATA BYTE @ HL
F21E 23    0445 INC      HL
F21F CD06F3 0446 CALL    PUT2HS       ;PRINT THE DATA IN HEX
F222 10F9  0447 DJNZ    DUMP2-$      ;REPEAT 16 TIMES
F224 E1    0448 POP      HL       ;RESTORE STARTING ADDRESS
F225 0610  0449 LD      B,16

```

```

F227 7E      0450 DUMP3: LD      A,(HL)          ;GET BACK DATA BYTE @ HL
F228 23      0451      INC      HL
F229 CBBF    0452      RES      7,A
F22B FE20    0453      CP       20H
F22D 3804    0454      JR       C,DUMP4-$
F22F FE7F    0455      CP       7FH
F231 3802    0456      JR       C,DUMP5-$
F233 3E2E    0457 DUMP4: LD      A,'.'          ;PRINT A DOT IF DATA 20 OR 7F
F235 CD4CF3  0458 DUMP5: CALL    OUTPUT          ;PRINT ASCII CHARACTER IN A
F238 10ED    0459      DJNZ     DUMP3-$
F23A CD32F3  0460      CALL     CRLFS
F23D C0       0461      RET      NZ              ;EXIT IF ESC REQ IS INDICATED
F23E 1B       0462      DEC      DE
F23F 7A       0463      LD       A,D
F240 B3       0464      OR       E
F241 20D1    0465      JR       NZ,DUMP-$
F243 C9       0466      RET
          0467 ;
          0468 ;
          0469 ;
          0470 ;
          0471 ;      -- READ FROM INPUT PORT COMMAND --
          0472 ;
F244 3D       0473 INCMD: DEC      A              ;CHECK IF PARAMETER COUNT=1
F245 37       0474      SCF
F246 C0       0475      RET      NZ
F247 4D       0476      LD       C,L              ;POINT C TO INPUT PORT
F248 CD32F3  0477 IN1:  CALL     CRLFS
F24B 79       0478      LD       A,C
F24C CD06F3  0479      CALL     PUT2HS
F24F ED78    0480      IN       A,(C)
F251 CD06F3  0481      CALL     PUT2HS
F254 CD3EF3  0482      CALL     ECHO
F257 FE0D    0483      CP       CR
F259 2806    0484      JR       Z,IN2-$
F25B FE2D    0485      CP       '-'
F25D 2804    0486      JR       Z,IN3-$
F25F B7       0487      OR       A
F260 C9       0488      RET
          0489
F261 0C       0490 IN2:  INC      C
F262 0C       0491      INC      C
F263 0D       0492 IN3:  DEC      C
F264 18E2    0493      JR       IN1-$
          0494 ;
          0495 ;
          0496 ;
          0497 ;      -- WRITE TO OUTPUT PORT COMMAND --
          0498 ;
F266 FE02    0499 OUTCMD: CP       2              ;CHECK IF PARAMETER COUNT=2
F268 37       0500      SCF
F269 C0       0501      RET      NZ
F26A 4D       0502      LD       C,L              ;POINT C TO OUTPUT PORT
F26B ED59    0503      OUT      (C),E            ;OUTPUT DATA PASSED IN E
F26D B7       0504      OR       A
F26E C9       0505      RET
          0506 ;
          0507 ;
          0508 ;*****
          0509 ;*
          0510 ;*      CONSOLE I/O PACKAGE AND UTILITY ROUTINES      *
          0511 ;*
          0512 ;*****
          0513 ;
          0514 ;
          0515 ;
F26F 41      0516 GETLIN: LD      B,C              ;SAVE MAX LINE LNGTH PARAMETR IN B

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F270	CD3EF3	0517	GLIN1:	CALL	ECHO	;GET A CHARACTER FROM THE CONSOLE
F273	FE0D	0518		CP	CR	;CHECK FOR CARRIAGE RETURN
F275	280E	0519		JR	Z, GLIN2-\$	
F277	FE08	0520		CP	'H'-64	;CHECK FOR CTL-H BACKSPACE
F279	280C	0521		JR	Z, GLIN4-\$	
F27B	FE20	0522		CP	' '	
F27D	D8	0523		RET	C	;OTHER CONTROL CHARS ARE ILLEGAL
F27E	77	0524		LD	(HL),A	
F27F	23	0525		INC	HL	;STORE CHARACTER IN BUFFER
F280	0D	0526		DEC	C	
F281	20ED	0527		JR	NZ, GLIN1-\$;GET ANOTHER IF THERE'S MORE ROOM
F283	37	0528		SCF		
F284	C9	0529		RET		;RETURN WITH CARRY=1 IF TOO
		0530				;MANY CHARACTERS ARE ENTERED
F285	77	0531	GLIN2:	LD	(HL),A	;PUT CARRIAGE RET ON END OF LINE
F286	C9	0532		RET		;RETURN WITH CARRY BIT=0
		0533				
F287	2B	0534	GLIN4:	DEC	HL	;DELETE LAST CHAR FROM BUFFER
F288	CD22F3	0535		CALL	PNEXT	
F28B	2008	0536		DEFB	' ', 'H'-64	;PRINT A SPACE TO OVERWRITE THE
F28D	04	0537		DEFB	EOT	; LAST CHAR, THEN DO A BACKSPACE
F28E	0C	0538		INC	C	
F28F	78	0539		LD	A,B	;MAKE SURE YOU'RE NOT TRYING TO
F290	91	0540		SUB	C	;BACKSPACE PAST THE START OF THE LINE
F291	30DD	0541		JR	NC, GLIN1-\$	
F293	C9	0542		RET		
		0543 ;				
		0544 ;				
		0545 ;				
F294	EDB1	0546	SEARCH:	CPIR		;SEARCH TBL @HL FOR MATCH WITH A
F296	C0	0547		RET	NZ	;EXIT NOW IF SEARCH FAILS
F297	09	0548		ADD	HL,BC	
F298	09	0549		ADD	HL,BC	;ADD RESIDUE FROM CPIR BYTECOUNT
F299	09	0550		ADD	HL,BC	; TO HL 3 TIMES TO GET POINTER
F29A	4E	0551		LD	C,(HL)	; TO ADDRESS PART OF TABLE ENTRY
F29B	23	0552		INC	HL	
F29C	46	0553		LD	B,(HL)	
F29D	C9	0554		RET		;EXIT WITH Z=1 TO INDICATE MATCH
		0555 ;				
		0556 ;				
		0557 ;				
		0558 ;				
F29E	010000	0559	PARAMS:	LD	BC,0	
F2A1	FD7E00	0560		LD	A,(IY+0)	
F2A4	FE0D	0561		CP	CR	;CHECK IF LINE TERMINATES
F2A6	2008	0562		JR	NZ, PARA2-\$; IMMEDIATELY WITH A RETURN
F2A8	AF	0563		XOR	A	
F2A9	C9	0564		RET		;RETURN WITH PARAM COUNT=0 IF SO
		0565				
F2AA	0C	0566	PARA1:	INC	C	
F2AB	0C	0567		INC	C	
F2AC	CB59	0568		BIT	3,C	
F2AE	37	0569		SCF		
F2AF	C0	0570		RET	NZ	;ERROR IF 4 NUMBERS ENTERED
F2B0	C5	0571	PARA2:	PUSH	BC	;SAVE PARAMETER COUNT
F2B1	CDD3F2	0572		CALL	GETHEX	;READ A NUMBER FROM LINE BUFFER
F2B4	C1	0573		POP	BC	
F2B5	D8	0574	PARA4:	RET	C	;ERROR IF RESULT OVER 16 BITS
F2B6	DD217CFF	0575		LD	IX,PARAM1	;POINT TO PARAMETER STORAGE AREA
F2BA	DD09	0576		ADD	IX,BC	;ADD PARAMETER COUNT IN BC
F2BC	DD7500	0577		LD	(IX+0),L	
F2BF	DD7401	0578		LD	(IX+1),H	;STORE DATA RETRND FROM 'GETHEX'
F2C2	FE20	0579		CP	' '	
F2C4	28E4	0580		JR	Z, PARA1-\$;GET ANOTHER ITEM IF SPACE
F2C6	FE2C	0581		CP	' ,'	
F2C8	28E0	0582		JR	Z, PARA1-\$;GET ANOTHER ITEM IF COMMA
F2CA	FE0D	0583		CP	CR	

F2CC	37	0584	SCF		;ELSE CHECK FOR CARRIAGE RETURN
F2CD	C0	0585	RET	NZ	; AND EXIT WITH CY=1 IF NOT
F2CE	79	0586	PAREND: LD	A,C	
F2CF	CB3F	0587	SRL	A	;A=COUNT OF NUMBERS ENTERED
F2D1	3C	0588	INC	A	
F2D2	C9	0589	RET		
		0590 ;			
		0591 ;			GETHEX CONVERTS ASCII TO BINARY AND DOES
		0592 ;			HIGH LIMIT CHECKS TO LESS THAN 17 BITS.
		0593 ;			CARRY SET ON ILLEGAL CONVERSION RESULT
		0594 ;			TERMINATING CHARACTER RETURNS IN A.
		0595 ;			HL RETURNS WITH 16 BIT BINARY INTEGER
		0596 ;			
F2D3	210000	0597	GETHEX: LD	HL,0	
F2D6	180B	0598	JR	GNUM3-\$	
		0599			
F2D8	0604	0600	GNUM1: LD	B,4	
F2DA	29	0601	GNUM2: ADD	HL,HL	;MULTIPLY RESULT BY 16
F2DB	D8	0602	RET	C	;RETURN IF IT OVERFLOWS 16 BITS
F2DC	10FC	0603	DJNZ	GNUM2-\$	
F2DE	5F	0604	LD	E,A	;APPEND NEW LOW ORDER DIGIT
F2DF	1600	0605	LD	D,0	;AND GET RESULT BACK INTO DE
F2E1	19	0606	ADD	HL,DE	
F2E2	D8	0607	RET	C	;RETURN IF OVERFLOW
F2E3	FD7E00	0608	GNUM3: LD	A,(IY+0)	;GET A CHARACTER FROM LINE INPUT
F2E6	FD23	0609	INC	IY	; BUFFER @ IY AND BUMP IY
F2E8	4F	0610	LD	C,A	
F2E9	CDF1F2	0611	CALL	ASCHEX	;CONVERT ASCII TO NUMERIC
F2EC	30EA	0612	JR	NC,GNUM1-\$	
F2EE	79	0613	LD	A,C	
F2EF	B7	0614	OR	A	
F2F0	C9	0615	RET		
		0616 ;			
		0617 ;			
F2F1	D630	0618	ASCHEX: SUB	'0'	
F2F3	D8	0619	RET	C	
F2F4	FE0A	0620	CP	10	
F2F6	3F	0621	CCF		
F2F7	D0	0622	RET	NC	
F2F8	D607	0623	SUB	7	
F2FA	FE0A	0624	CP	10	
F2FC	D8	0625	RET	C	
F2FD	FE10	0626	CP	16	
F2FF	3F	0627	CCF		
F300	C9	0628	RET		
		0629 ;			
		0630 ;			
		0631 ;			
F301	7C	0632	PUT4HS: LD	A,H	
F302	CD0DF3	0633	CALL	PUT2HX	
F305	7D	0634	LD	A,L	
F306	CD0DF3	0635	PUT2HS: CALL	PUT2HX	
F309	CD38F3	0636	CALL	SPACE	
F30C	C9	0637	RET		
		0638 ;			
		0639 ;			
F30D	F5	0640	PUT2HX: PUSH	AF	
F30E	1F	0641	RRA		
F30F	1F	0642	RRA		
F310	1F	0643	RRA		
F311	1F	0644	RRA		
F312	CD16F3	0645	CALL	PUTNIB	
F315	F1	0646	POP	AF	
F316	E60F	0647	PUTNIB: AND	00001111B	
F318	C690	0648	ADD	A,90H	
F31A	27	0649	DAA		
F31B	CE40	0650	ADC	A,40H	

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F31D	27	0651	DAA		
F31E	CD4CF3	0652	CALL	OUTPUT	
F321	C9	0653	RET		
		0654 ;			
		0655 ;			
		0656 ;		PMSG PRINTS THE STRING OF ASCII CHARACTERS	
		0657 ;		POINTED TO BY THE RELATIVE ADDRESS IN DE	
		0658 ;		UNTIL AN EOT IS ENCOUNTERED IN THE STRING.	
		0659 ;			
0004		0660 EOT	EQU	04H	
000D		0661 CR	EQU	0DH	
000A		0662 LF	EQU	0AH	
		0663 ;			
		0664			
F322	E3	0665 PNEXT:	EX	(SP),HL	
F323	CD28F3	0666	CALL	PMSG	
F326	E3	0667	EX	(SP),HL	
F327	C9	0668	RET		
		0669 ;			
F328	7E	0670 PMSG:	LD	A,(HL)	
F329	23	0671	INC	HL	
F32A	FE04	0672	CP	EOT	
F32C	C8	0673	RET	Z	
F32D	CD4CF3	0674	CALL	OUTPUT	
F330	18F6	0675	JR	PMSG-\$	
		0676 ;			
		0677 ;			
		0678 ;		CRLFS OUTPUTS A RETURN-LINEFEED-SPACE	
		0679 ;		TO THE CONSOLE DEVICE	
		0680 ;			
F332	CD22F3	0681 CRLFS:	CALL	PNEXT	
F335	0D0A04	0682	DEFB	CR,LF,EOT	
F338	3E20	0683 SPACE:	LD	A,' '	
F33A	CD4CF3	0684	CALL	OUTPUT	
F33D	C9	0685	RET		
		0686 ;			
		0687 ;			
		0688 ;			
		0689 ;		ECHO INPUTS ONE CHARACTER FROM THE CONSOLE	
		0690 ;		DEVICE, PRINTS IT ON THE CONSOLE OUTPUT AND	
		0691 ;		THEN RETURNS IT IN REGISTER A WITH BIT 7 RESET	
		0692 ;			
		0693 ;		OUTPUT PRINTS THE CHARACTER IN REGISTER A ON	
		0694 ;		THE CONSOLE OUTPUT DEVICE AND THEN DOES A CHECK	
		0695 ;		FOR CONSOLE INPUT TO FREEZE OR ABORT OUTPUT.	
		0696 ;			
		0697			
F33E	CD09F0	0698 ECHO:	CALL	CONIN	;INPUT A CHARACTER AND ECHO IT
F341	F5	0699	PUSH	AF	
F342	CD0CF0	0700	CALL	CONOUT	
F345	F1	0701	POP	AF	
F346	FE5B	0702	CP	'Z'+1	
F348	D8	0703	RET	C	
F349	D620	0704	SUB	32	;CONVERT UPPER CASE TO LOWER CASE
F34B	C9	0705	RET		
		0706 ;			
		0707 ;			
		0708 ;			
F34C	CD0CF0	0709 OUTPUT:	CALL	CONOUT	
F34F	CD06F0	0710	CALL	CONST	;SEE IF CONSOLE INPUT IS PENDING
F352	280F	0711	JR	Z,OUTP2-\$	
F354	CD09F0	0712	CALL	CONIN	
F357	FE0D	0713	CP	CR	;SEE IF CARRIAGE RET WAS TYPED
F359	2805	0714	JR	Z,OUTP1-\$	
F35B	CD09F0	0715	CALL	CONIN	;WAIT FOR ANOTHER INPUT CHAR
F35E	1803	0716	JR	OUTP2-\$; THEN RETURN TO CALLING ROUTINE
		0717			

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

F360 3284FF 0718 OUTP1: LD      (ESCFLG),A      ;SET ESC FLAG TO NON-ZERO VALUE
F363 3A84FF 0719 OUTP2: LD      A,(ESCFLG)
F366 B7      0720      OR      A              ;RETURN CURRENT STATUS OF ESCAPE
F367 C9      0721      RET                    ; FLAG TO CALLING ROUTINE
          0722 ;
          0723 ;
          0724 ;
          0725      INCLUDE INTRSRV.ASM
          0726 ;*****
          0727 ;*
          0728 ;*      INTERRUPT SERVICE ROUTINES FOR KEYBOARD      *
          0729 ;*      INPUT AND REAL-TIME CLOCK FUNCTIONS          *
          0730 ;*
          0731 ;*      XEROX 820 VERSION 1.0      10-OCT-80      *
          0732 ;*      VERSION 2.0      21-JULY-81      *
          0733 ;*
          0734 ;*****
          0735 ;
          0736 ;
          0737 ;
          0738 ;
F368 3A30FF 0739 KBDST: LD      A,(FIFCNT)      ;GET INPUT FIFO BYTECOUNT
F36B B7      0740      OR      A              ;TEST IF EQUAL ZERO
F36C C8      0741      RET      Z              ;EXIT WITH A=0 IF QUEUE IS EMPTY
F36D 3EFF    0742      LD      A,255
F36F C9      0743      RET                    ;ELSE SET A=255 TO INDICATE DATA RDY
          0744 ;
          0745 ;
          0746 ;
F370 CD68F3 0747 KBDIN: CALL    KBDST
F373 28FB    0748      JR      Z,KBDIN-$      ;LOOP UNTIL KEYBOARD INPUT READY
F375 E5      0749      PUSH   HL
F376 CD8FF3 0750      CALL    REMOVE          ;GET CHARACTER FROM INPUT QUEUE
F379 E1      0751      POP     HL
F37A C9      0752      RET
          0753 ;
          0754 ;
          0755 ;
          0756 ;
          0757 ;
F37B EE20    0758      XOR     00100000B      ;ELSE TOGGLE BIT 5 OF THE CHAR
F37D 4F      0759 STASH3: LD      C,A
F37E 2130FF 0760      LD      HL,FIFCNT      ;BUMP INPUT FIFO CHARACTER COUNT
F381 7E      0761      LD      A,(HL)
F382 3C      0762      INC     A
F383 FE10    0763      CP      16
F385 D0      0764      RET      NC            ;EXIT NOW IF FIFO IS FULL
F386 77      0765      LD      (HL),A        ; ELSE INCREMENT FIFO COUNT
F387 2131FF 0766      LD      HL,FIFIN      ;POINT HL TO FIFO INPUT OFFSET
F38A CD96F3 0767      CALL    INDEX
F38D 71      0768      LD      (HL),C        ;STORE CHARACTER IN FIFO @ HL
F38E C9      0769      RET
          0770 ;
          0771 ;
          0772 ;
          0773 ;
F38F 2130FF 0774 REMOVE: LD      HL,FIFCNT
F392 35      0775      DEC     (HL)
F393 2132FF 0776      LD      HL,FIFOUT      ;POINT HL TO FIFO OUTPUT OFFSET
F396 7E      0777 INDEX: LD      A,(HL)
F397 3C      0778      INC     A
F398 E60F    0779      AND     00001111B      ;INCREMENT FIFO POINTER
F39A 77      0780      LD      (HL),A        ; MODULO 16 AND REPLACE
F39B 2120FF 0781      LD      HL,FIFO
F39E 85      0782      ADD     A,L            ;INDEX INTO FIFO BY OFFSET IN A
F39F 6F      0783      LD      L,A
F3A0 7E      0784      LD      A,(HL)

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

F3A1  C9          0785      RET
                    0786 ;
                    0787 ;
                    0788      SOFTWARE DISK MOTOR TURN-OFF TIMER ROUTINE
                    0789 ;
F3A2  2169FF     0790 DSKTMR: LD      HL,MOTOR      ;DECREMENT DISK TURN-OFF TIMER
F3A5  35          0791      DEC      (HL)
F3A6  C0          0792      RET      NZ              ;EXIT IF NOT TIMED OUT YET
F3A7  DB1C       0793      IN       A,(BITDAT)
F3A9  E6F8       0794      AND      11111000B      ;DISABLE ALL DRIVE SELECTS AND
F3AB  D31C       0795      OUT      (BITDAT),A      ; TURN OFF THE SPINDLE MOTORS
F3AD  C9          0796      RET
                    0797 ;
                    0798 ;
                    0799 ;
                    0800 ;
                    0801      -- INTERRUPT SERVICE ROUTINE FOR PARALLEL KEYBOARD --
                    0802 ;
F3AE  ED7335FF   0803 KEYSRV: LD      (SPSAVE),SP    ;SAVE USER STACK POINTER AND
F3B2  3157FF     0804      LD      SP,TMPSTK+32      ; SWITCH TO LOCAL STACK
F3B5  E5          0805      PUSH    HL
F3B6  D5          0806      PUSH    DE
F3B7  C5          0807      PUSH    BC
F3B8  F5          0808      PUSH    AF              ;SAVE MACHINE STATE
F3B9  DB1E       0809      IN      A,(KBDDAT)        ;READ KEYBOARD INPUT PORT
F3BB  2F         0810      CPL
F3BC  E67F       0811      AND      01111111B
F3BE  CD7DF3     0812      CALL    STASH3
F3C1  F1         0813      POP     AF
F3C2  C1         0814      POP     BC
F3C3  D1         0815      POP     DE
F3C4  E1         0816      POP     HL
F3C5  ED7B35FF   0817      LD      SP,(SPSAVE)
F3C9  FB         0818      EI                      ;RE-ENABLE INTERRUPTS AND RETURN
F3CA  ED4D       0819      RETI
                    0820 ;
                    0821 ;
                    0822 ;
                    0823      -- INTERRUPT SERVICE ROUTINE FOR ONE SECOND TIMER --
                    0824 ;
F3CC  ED7335FF   0825 TIMER:  LD      (SPSAVE),SP    ;SAVE USER STACK POINTER AND
F3D0  3157FF     0826      LD      SP,TMPSTK+32      ; SWITCH TO LOCAL STACK
F3D3  E5          0827      PUSH    HL
F3D4  D5          0828      PUSH    DE
F3D5  C5          0829      PUSH    BC
F3D6  F5          0830      PUSH    AF
F3D7  CDA2F3     0831      CALL    DSKTMR            ;GO SRVCE THE DISK TURN OFF TIMER
F3DA  F1         0832      POP     AF
F3DB  C1         0833      POP     BC
F3DC  D1         0834      POP     DE
F3DD  E1         0835      POP     HL
F3DE  ED7B35FF   0836      LD      SP,(SPSAVE)
F3E2  FB         0837      EI                      ;RE-ENABLE INTERRUPTS AND RETURN
F3E3  ED4D       0838      RETI
                    0839 ;
                    0840 ;
                    0841 ;
F3E5  ED7335FF   0842 MILLI:  LD      (SPSAVE),SP    ;SAVE USER STACK POINTER AND
F3E9  3157FF     0843      LD      SP,TMPSTK+32      ; SWITCH TO LOCAL STACK
F3EC  E5          0844      PUSH    HL
F3ED  F5          0845      PUSH    AF
F3EE  2A70FF     0846      LD      HL,(INDTMR)
F3F1  2B         0847      DEC     HL              ;DECREMENT INDEX PERIOD TIMER
F3F2  2270FF     0848      LD      (INDTMR),HL
F3F5  F1         0849      POP     AF
F3F6  E1         0850      POP     HL
F3F7  ED7B35FF   0851      LD      SP,(SPSAVE)

```

```

F3FB FB      0852      EI
F3FC ED4D    0853      RETI
                0854 ;
                0855 ;
                0856 ;
                0857 ;
                0858 ;
                0859 ;          POLLED MODE I/O ROUTINES FOR SIO CHANEL B
                0860 ;
F3FE DB07    0861 SIOST: IN      A,(SIOCPB)      ;GET SIO STATUS REGISTER
F400 E601    0862      AND      00000001B
F402 C8      0863      RET      Z              ;ACC=0 IF NO DATA AVAILABLE
F403 3EFF    0864      LD      A,255
F405 C9      0865      RET
                0866 ;
                0867 ;
F406 CDFEF3  0868 SIOIN: CALL    SIOST          ;TEST CONSOLE STATUS
F409 28FB    0869      JR      Z,SIOIN-$        ;LOOP UNTIL DATA IS
F40B DB05    0870      IN      A,(SIODPB)        ; READY AT SIO DATA PORT
F40D E67F    0871      AND      01111111B
F40F C9      0872      RET
                0873 ;
                0874 ;
F410 F5      0875 SIOOUT: PUSH    AF
F411 DB07    0876 SIOX1: IN      A,(SIOCPB)
F413 E604    0877      AND      00000100B        ;TEST TBE STATUS BIT
F415 28FA    0878      JR      Z,SIOX1-$
F417 DB07    0879 SIOX2: IN      A,(SIOCPB)        ;TEST DCD STATUS BIT
F419 E608    0880      AND      08H
F41B 28FA    0881      JR      Z,SIOX2-$        ;LOOP UNTIL BIT SET VER. 2.0
F41D F1      0882      POP      AF
F41E D305    0883      OUT      (SIODPB),A        ;OUTPUT DATA TO SIO
F420 C9      0884      RET
                0885 ;
                0886 ;
                0887 ;
                0888 ;
                0889      INCLUDE CRTOUT.ASM
0890 ;*****
0891 ;*
0892 ;*      MEMORY-MAPPED CRT OUTPUT DRIVER
0893 ;*
0894 ;*
0895 ;*****
0896 ;
0897 ;
0030          0898 CRTBAS EQU      CRTMEM.SHR.8      ;STARTING PAGE# OF 3K CRT SPACE
003C          0899 CRTTOP EQU      CRTMEM+3072.SHR.8 ;ENDING PAGE# OF CRT SPACE
                0900 ;
                0901 ;
F421 E5      0902 CRTOUT: PUSH    HL
F422 D5      0903      PUSH    DE
F423 C5      0904      PUSH    BC
                0905 ;      RES      7,A          ;ALLOW BLINKING MODE
F424 4F      0906      LD      C,A
F425 F3      0907      DI
F426 ED7335FF 0908      LD      (SPSAVE),SP
F42A 3157FF  0909      LD      SP,TMPSTK+32        ;POINT SP TO TOP OF LOCAL STACK
F42D DB1C    0910      IN      A,(BITDAT)
F42F CBFF    0911      SET      7,A          ;SELECT ROM/CRT MEMORY BANK
F431 D31C    0912      OUT      (BITDAT),A
                0913 ;
                0914 ;      FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN
                0915 ;
F433 2176FF  0916      LD      HL,CHRSAB        ;GET CHAR NOW OVERLAYED BY CURSOR
F436 46      0917      LD      B,(HL)
F437 2A74FF  0918      LD      HL,(CURSOR)        ;LOAD HL WITH CURSOR POINTER

```

F43A	7C	0919	LD	A,H	
F43B	E60F	0920	AND	00001111B	;A LITTLE INSURANCE THAT HL CAN'T
F43D	F630	0921	OR	CRTBAS	; EVER POINT OUTSIDE THE CRT MEMORY
F43F	67	0922	LD	H,A	
F440	70	0923	LD	(HL),B	;REMOVE CURSOR BY RESTORING CHAR
		0924 ;			
		0925 ;			
		0926 ;			
		0927	CALL	OUTCH	
F441	CD64F4	0928 ;			
		0929 ;			
		0930 ;			
		0931	LD	A,(HL)	;GET CHAR AT NEW CURSOR LOCATION
F444	7E	0932	LD	(CHRSAB),A	;SAVE FOR NXT TME 'CRTOUT' IS CLD
F445	3276FF	0933	CP	' '	;TEST IF CHARACTER IS A SPACE
F448	FE20	0934	SET	7,A	;THEN TURN ON BIT 7 TO ENBL BLNK
F44A	CBFF	0935	JR	NZ,CRT2-\$;JUMP IF CHARACTER IS NON-BLANK
F44C	2003	0936	LD	A,(CSRCHR)	; ELSE GET CHAR USED FOR CURSOR
F44E	3A77FF	0937	LD	(HL),A	;STORE CHAR IN A AS CURSOR MARK
F451	77	0938	LD	(CURSOR),HL	;SAVE HL AS CURSOR POINTER
F452	2274FF	0939			
		0940	LD	SP,(SPSAVE)	
F455	ED7B35FF	0941	IN	A,(BITDAT)	
F459	DB1C	0942	RES	7,A	;SWITCH BACK THE LOWER 16K OF RAM
F45B	CBBF	0943	OUT	(BITDAT),A	
F45D	D31C	0944	EI		;INTERRUPTS ARE SAFE AGAIN
F45F	FB	0945	POP	BC	
F460	C1	0946	POP	DE	
F461	D1	0947	POP	HL	
F462	E1	0948	RET		
F463	C9	0949 ;			
		0950 ;			
		0951 ;			
F464	1179FF	0952	OUTCH: LD	DE,LEADIN	
F467	1A	0953	LD	A,(DE)	;GET LEAD-IN SEQUENCE STATE
F468	B7	0954	OR	A	
F469	C26FF5	0955	JP	NZ,MULTI	;JUMP IF IN A LEAD-IN SEQUENCE
F46C	79	0956	LD	A,C	; ELSE PROCESS CHARACTER IN C
F46D	FE20	0957	CP	' '	
F46F	380F	0958	JR	C,CONTRL-\$;JUMP IF A CONTROL CHARACTER
F471	71	0959	DISPLA: LD	(HL),C	; ELSE STORE DISPLAYABLE CHAR
F472	23	0960	INC	HL	; AND ADVANCE POINTER TO NEXT COL
F473	7D	0961	LD	A,L	
F474	E67F	0962	AND	01111111B	;EXTRACT COLUMN# FROM HL
F476	FE50	0963	CP	80	
F478	D8	0964	RET	C	;EXIT IF NOT PAST COLUMN 79
F479	CDE6F4	0965	CALL	RETURN	; ELSE DO AUTOMATIC CARRIAGE RET
F47C	CD41F5	0966	CALL	LFEED	; AND LINEFEED
F47F	C9	0967	RET		
		0968 ;			
		0969 ;			
		0970 ;			
F480	E5	0971	CONTRL: PUSH	HL	
F481	218EF4	0972	LD	HL,CTLTAB	;SEARCH FOR CONTROL CHARACTER
F484	010D00	0973	LD	BC,CTLSIZ/3	; HANDLING SUBROUTINE IN TABLE
F487	CD94F2	0974	CALL	SEARCH	
F48A	E1	0975	POP	HL	
F48B	C0	0976	RET	NZ	;EXIT IF NOT IMPLEMENTED
F48C	C5	0977	PUSH	BC	
F48D	C9	0978	RET		;DO SNEAKY JUMP TO PRESERVE REGS
		0979			
F48E	1F	0980	CTLTAB: DEFB	' '-64	
F48F	1E	0981	DEFB	' '-64	
F490	1B	0982	DEFB	' '-64	
F491	1A	0983	DEFB	'Z'-64	
F492	18	0984	DEFB	'X'-64	
F493	11	0985	DEFB	'Q'-64	

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F494	OD	0986	DEFB	'M'-64	
F495	OC	0987	DEFB	'L'-64	
F496	OB	0988	DEFB	'K'-64	
F497	OA	0989	DEFB	'J'-64	
F498	O9	0990	DEFB	'I'-64	
F499	O8	0991	DEFB	'H'-64	
F49A	O7	0992	DEFB	'G'-64	
		0993			
F49B	DBF4	0994	DEFW	BELL	;CTL-G IS THE BELL
F49D	BDF4	0995	DEFW	BAKSPC	;CTL-H IS CURSOR LEFT
F49F	CBF4	0996	DEFW	TAB	;CTL-I IS TAB
F4A1	41F5	0997	DEFW	LFEED	;CTL-J IS CURSOR DOWN
F4A3	2BF5	0998	DEFW	UPCSR	;CTL-K IS CURSOR UP
F4A5	C3F4	0999	DEFW	FORSPC	;CTL-L IS CURSOR RIGHT
F4A7	E6F4	1000	DEFW	RETURN	;CTL-M IS CARRIAGE RETURN
F4A9	10F5	1001	DEFW	CLREOS	;CTL-Q IS CLEAR TO END-OF-SCREEN
F4AB	02F5	1002	DEFW	CLREOL	;CTL-X IS CLEAR TO END-OF-LINE
F4AD	EBF4	1003	DEFW	CLRSCN	;CTL-Z IS CLEAR SCREEN
F4AF	B5F4	1004	DEFW	ESCAPE	;CTL-, IS ESCAPE
F4B1	6BF5	1005	DEFW	HOMEUP	;CTL- IS HOME UP
F4B3	B9F4	1006	DEFW	STUFF	;CTL- IS DISPLAY CONTROL CHARS
		1007			
0027		1008	CTLSIZ	EQU	\$-CTLTAB
		1009			
		1010			
F4B5	3E01	1011	ESCAPE:	LD	A,1
F4B7	12	1012		LD	(DE),A
F4B8	C9	1013		RET	;SET LEAD-IN SEQUENCE STATE
		1014			; FOR XY CURSOR POSITIONING MODE
		1015			
F4B9	3E04	1016	STUFF:	LD	A,4
F4BB	12	1017		LD	(DE),A
F4BC	C9	1018		RET	;SET LEAD-IN SEQUENCE STATE
		1019			; FOR CONTROL CHAR OUTPUT MODE
		1020			
F4BD	7D	1021	BAKSPC	LD	A,L
F4BE	E67F	1022		AND	01111111B
F4C0	C8	1023		RET	Z
F4C1	2B	1024		DEC	HL
F4C2	C9	1025		RET	
		1026			
		1027			
F4C3	7D	1028	FORSPC:	LD	A,L
F4C4	E67F	1029		AND	01111111B
F4C6	FE4F	1030		CP	79
F4C8	D0	1031		RET	NC
F4C9	23	1032		INC	HL
F4CA	C9	1033		RET	
		1034			;CHECK FOR LEFT MARGIN
		1035			
F4CB	110800	1036	TAB:	LD	DE,8
F4CE	7D	1037		LD	A,L
F4CF	E678	1038		AND	01111000B
F4D1	83	1039		ADD	A,E
F4D2	FE50	1040		CP	80
F4D4	D0	1041		RET	NC
F4D5	7D	1042		LD	A,L
F4D6	E6F8	1043		AND	11111000B
F4D8	6F	1044		LD	L,A
F4D9	19	1045		ADD	HL,DE
F4DA	C9	1046		RET	
		1047			
		1048			
F4DB	DB1C	1049	BELL:	IN	A,(BITDAT)
F4DD	CBEF	1050		SET	5,A
F4DF	D31C	1051		OUT	(BITDAT),A
F4E1	CBAF	1052		RES	5,A
					;TOGGLE BIT 5 OF SYSTEM PIO TO
					; TRIGGER BELL HARDWARE TO SOUND

F4E3	D31C	1053	OUT	(BITDAT),A	
F4E5	C9	1054	RET		
		1055 ;			
		1056 ;			
F4E6	7D	1057	RETURN: LD	A,L	
F4E7	E680	1058	AND	10000000B	
F4E9	6F	1059	LD	L,A	;MOVE CURSOR POINTER BACK
F4EA	C9	1060	RET		; TO START OF LINE
		1061 ;			
		1062 ;			
F4EB	210030	1063	CLRSCN: LD	HL,CRTMEM	
F4EE	E5	1064	PUSH	HL	
F4EF	110130	1065	LD	DE,CRTMEM+1	
F4F2	01000C	1066	LD	BC,24*128	
F4F5	3620	1067	LD	(HL),' '	
F4F7	EDB0	1068	LDIR		;FILL CRT MEMORY WITH SPACES
F4F9	E1	1069	POP	HL	;POINT TO HOME CURSOR POSITION
F4FA	3E17	1070	LD	A,23	
F4FC	3278FF	1071	LD	(BASE),A	;MAKE BASE LINE# BE 23 AND
F4FF	D314	1072	OUT	(SCROLL),A	; STORE IN SCROLL REGISTER
F501	C9	1073	RET		
		1074 ;			
		1075 ;			
F502	E5	1076	CLREOL: PUSH	HL	;SAVE CURSOR POINTER
F503	7D	1077	LD	A,L	
F504	E67F	1078	AND	01111111B	;GET COLUMN# COMPONENT OF
F506	4F	1079	LD	C,A	; CURSOR POINTER INTO C
F507	3E50	1080	LD	A,80	;CALCULATE HOW MANY CHARACTERS
F509	91	1081	SUB	C	; REMAIN ON CURRENT LINE
F50A	47	1082	LD	B,A	
F50B	CD65F5	1083	CALL	CLR	;CLEAR REST OF LINE @ HL
F50E	E1	1084	POP	HL	
F50F	C9	1085	RET		
		1086 ;			
		1087 ;			
F510	CD02F5	1088	CLREOS: CALL	CLREOL	;CLEAR REMAINDER OF CURRENT ROW
F513	E5	1089	PUSH	HL	
F514	3A78FF	1090	LD	A,(BASE)	
F517	4F	1091	LD	C,A	;COPY BASE SCREEN ROW# TO C
F518	7D	1092	CLRS1: LD	A,L	
F519	17	1093	RLA		
F51A	7C	1094	LD	A,H	
F51B	17	1095	RLA		;GET ROW# COMPONENT OF HL INTO A
F51C	E61F	1096	AND	00011111B	
F51E	B9	1097	CP	C	;SEE IF HL IS AT BTM ROW OF SCRNM
F51F	2808	1098	JR	Z,CLRS2-\$; AND LEAVE CLEAR LOOP IF SO
F521	CD36F5	1099	CALL	DNCSR	;ELSE POINT HL TO NEXT ROW DOWN
F524	CD5FF5	1100	CALL	CLRLIN	; AND FILL THAT LINE WITH SPACES
F527	18EF	1101	JR	CLRS1-\$	
		1102			
F529	E1	1103	CLRS2: POP	HL	;RESTORE ORIGINAL CURSOR POINTER
F52A	C9	1104	RET		
		1105 ;			
		1106 ;			
F52B	1180FF	1107	UPCSR: LD	DE,-128	;SUBTRACT 1 FROM ROW# COMPONENT
F52E	19	1108	ADD	HL,DE	; OF CURSOR POINTER IN HL
F52F	7C	1109	LD	A,H	
F530	FE30	1110	CP	CRTBAS	;CHECK FOR UNDERFLOW OF POINTER
F532	D0	1111	RET	NC	
F533	263B	1112	LD	H,CRTTOP-1	;WRAP CURSOR AROUND MODULO 3K
F535	C9	1113	RET		
		1114 ;			
		1115 ;			
F536	118000	1116	DNCSR: LD	DE,128	;ADD 1 TO ROW# COMPONENT
F539	19	1117	ADD	HL,DE	; OF CURSOR POINTER IN HL
F53A	7C	1118	LD	A,H	
F53B	FE3C	1119	CP	CRTTOP	;CHECK FOR OVERFLOW OF POINTER

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F53D	D8	1120	RET	C	
F53E	2630	1121	LD	H,CRTBAS	;RESET POINTER MODULO 128*24
F540	C9	1122	RET		
		1123 ;			
		1124 ;			
		1125 ;			
F541	7D	1126	LFEED: LD	A,L	
F542	17	1127	RLA		
F543	7C	1128	LD	A,H	
F544	17	1129	RLA		;EXTRACT ROW# COMPONENT OF HL
F545	E61F	1130	AND	00011111B	
F547	4F	1131	LD	C,A	;COPY ROW# INTO C FOR SCROLL TEST
F548	CD36F5	1132	CALL	DNCSR	;MOVE CURSOR TO NEXT ROW DOWN
F54B	3A78FF	1133	LD	A,(BASE)	;TEST IF CURSOR WAS ON BOTTOM ROW
F54E	B9	1134	CP	C	; OF SCREEN BEFORE MOVING DOWN
F54F	C0	1135	RET	NZ	;EXIT IF NOT AT BOTTOM
		1136			
F550	E5	1137	PUSH	HL	;ELSE PREP TO SCROLL SCREEN UP
F551	CD5FF5	1138	CALL	CLRLIN	;FILL NEW BOTTOM LINE WITH SPACES
F554	29	1139	ADD	HL,HL	
F555	7C	1140	LD	A,H	;GET ROW# COMPONENT OF HL INTO A
F556	E61F	1141	AND	00011111B	
F558	3278FF	1142	LD	(BASE),A	;STORE NEW BASE LINE#
F55B	D314	1143	OUT	(SCROLL),A	;NOW SCROLL UP NEW BLNK BTM LINE
F55D	E1	1144	POP	HL	
F55E	C9	1145	RET		
		1146 ;			
		1147 ;			
F55F	7D	1148	CLRLIN: LD	A,L	
F560	E680	1149	AND	10000000B	;POINT HL TO FIRST COLUMN OF ROW
F562	6F	1150	LD	L,A	
F563	0650	1151	LD	B,80	
F565	3620	1152	CLR: LD	(HL),' '	;STORE ASCII SPCS AT ADDRS IN HL
F567	23	1153	INC	HL	; AND INCREMENT HL
F568	10FB	1154	DJNZ	CLR-\$;REPEAT NMBR OF TIMES GIVEN BY B
F56A	C9	1155	RET		
		1156 ;			
		1157 ;			
F56B	0E20	1158	HOMEUP: LD	C,' '	;FAKE-OUT CURSOR ADDRSGING ROUTINE
F56D	1817	1159	JR	SETROW-\$; TO DO HOMEUP ALMOST FOR FREE
		1160 ;			
		1161 ;			
F56F	EB	1162	MULTI: EX	DE,HL	;UNCONDITNLY RESET THE LEAD-IN
F570	3600	1163	LD	(HL),0	; STATE TO ZERO BEFORE GOING ON
F572	EB	1164	EX	DE,HL	
F573	FE01	1165	CP	1	
F575	2008	1166	JR	NZ,M2TST-\$	
F577	79	1167	SETXY: LD	A,C	;GET SECOND CHAR OF SEQUENCE
F578	FE3D	1168	CP	'='	
F57A	C0	1169	RET	NZ	;ABORT SEQUENCE IF NOT '='
F57B	3E02	1170	LD	A,2	
F57D	12	1171	LD	(DE),A	;MAKE LEADIN=2 NEXT TIME
F57E	C9	1172	RET		
		1173			
F57F	FE02	1174	M2TST: CP	2	
F581	2019	1175	JR	NZ,M3TST-\$	
F583	3E03	1176	LD	A,3	
F585	12	1177	LD	(DE),A	;MAKE LEADIN=3 NEXT TIME
F586	3A78FF	1178	SETROW: LD	A,(BASE)	;ARRIVE HERE ON THIRD CHARACTER
F589	81	1179	ADD	A,C	; OF ESC,'=',ROW,COL SEQUENCE
F58A	D61F	1180	SUB	'-1	
F58C	D618	1181	SETR2: SUB	24	
F58E	30FC	1182	JR	NC,SETR2-\$;MAKE SURE ROW# IS BTWN 0 AND 23
F590	C618	1183	ADD	A,24	
F592	F660	1184	OR	CRTMEM.SHR.7	;MERGE IN MSB'S OF CRT MEMORY
F594	67	1185	LD	H,A	
F595	2E00	1186	LD	L,0	

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

F597 CB3C      1187      SRL      H
F599 CB1D      1188      RR       L
F59B C9        1189      RET
1190
F59C FE03      1191 M3TST: CP      3
F59E 200C      1192      JR       NZ,M4TST-$
F5A0 79        1193 SETCOL: LD      A,C      ;ARRIVE HERE ON FOURTH CHARACTER
F5A1 D620      1194      SUB      ' '      ; OF ESC,'=',ROW,COL SEQUENCE
F5A3 D650      1195 SETC2: SUB      80
F5A5 30FC      1196      JR       NC,SETC2-$      ;MAKE SURE COL# IS BETWEEN 0 & 79
F5A7 C650      1197      ADD      A,80
F5A9 B5        1198      OR       L      ;MERGE IN COL# WITH L
F5AA 6F        1199      LD       L,A
F5AB C9        1200      RET
1201
F5AC CD71F4    1202 M4TST: CALL    DISPLA      ;DISPLAY THE CONTROL CHARACTER
F5AF C9        1203      RET      ; PASSED IN C
1204 ;
1205 ;
1206 ;
1207 ;
1208      INCLUDE DISKIO.ASM
1209 ;*****
1210 ;*
1211 ;*      DISK INPUT/OUTPUT DRIVER SUBROUTINE PACKAGE      *
1212 ;*      FOR WESTERN DIGITAL 1771 DISK CONTROLLER          *
1213 ;*
1214 ;*      VERSION 2.0 FOR SA400, SA800, SA450 DISC DRIVE      *
1215 ;*      JULY 28, 1981                                       *
1216 ;*
1217 ;*****
1218 ;
1219 ;
1220 ;      EQUATES FOR DISK CONTROLLER PORTS AND COMMAND CODES
1221 ;
0010      1222 STSREG EQU      WD1771+0      ;STATUS REGISTER
0010      1223 CMDREG EQU      WD1771+0      ;COMMAND REGISTER
0011      1224 TRKREG EQU      WD1771+1      ;TRACK REGISTER
0012      1225 SECREG EQU      WD1771+2      ;SECTOR REGISTER
0013      1226 DATREG EQU      WD1771+3      ;DATA REGISTER
1227 ;
0088      1228 RDCMD EQU      10001000B      ;READ COMMAND
00A8      1229 WRTCMD EQU      10101000B      ;WRITE COMMAND
001C      1230 SKCMD EQU      00011100B      ;SEEK COMMAND
00D0      1231 FINCMD EQU      11010000B      ;FORCE INTR COMMAND
000C      1232 RSTCMD EQU      00001100B      ;RESTORE COMMAND
0004      1233 HLOAD EQU      00000100B      ;RD/WRT HEAD LOAD ENABLE
1234 ;
00C9      1235 RET EQU      0C9H      ;SUBROUTINE RETURN INSTR OPCODE
0066      1236 NMIVEC EQU      0066H      ;THE NON-MASKABLE INTERRUPT IS
1237      ;USED FOR DATA SYNCRONIZATN BTWN
1238      ;THE Z-80 AND 1771 DISK CONTROLLER
1239 ;
000B      1240 RECNT EQU      11      ;NUMBER OF ERROR RETRY
1241 ;
1242 ;
F5B0 OC      1243 SELECT: INC      C      ;MAKE DRIVE ID FROM 1 TO 4
F5B1 79      1244      LD       A,C
F5B2 FE05      1245      CP       5      ; CHECK FOR MAXIMUM VALID#
F5B4 D0      1246      RET      NC      ;ERROR IF NUMBER 5
F5B5 FE03      1247      CP       3      ;TEST IF DRIVE SELECT IN SIDE 1
F5B7 3802      1248      JR       C,RSTMR-$      ;NO, KEEP DRIVE ID 1 AND 2
F5B9 OC      1249      INC      C
F5BA OC      1250      INC      C      ;YES,MAKE DRIVE ID 5 OR 6
F5BB CDC2F6    1251 RSTMR: CALL    RSTMR      ;RESET MTR TIMER & GET PORT DATA
F5BE 47      1252      LD       B,A      ;SAVE CURRENT DRIVE SELECT DATA
F5BF E6F8      1253      AND      11111000B

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F5C1	B1	1254	OR	C	;MERGE IN NEW DRIVE UNIT#
F5C2	CDDBF6	1255	CALL	TURNON	;SEE IF NEW DRIVE IS READY
F5C5	2007	1256	JR	NZ,SEL2-\$; AND CONTINUE IF ITS READY
F5C7	78	1257	LD	A,B	;ELSE GET BACK PREV DRIVE SELECT
F5C8	D31C	1258	OUT	(BITDAT),A	
F5CA	3E80	1259	LD	A,10000000B	
F5CC	B7	1260	OR	A	;RETURN DRIVE NOT READY INDICATN
F5CD	C9	1261	RET		
		1262			
F5CE	215FFF	1263	SEL2: LD	HL,UNIT	;POINT HL TO DRIVE SELECT DATA
F5D1	7E	1264	LD	A,(HL)	;LOAD A WITH CURRENT UNIT#
F5D2	71	1265	LD	(HL),C	; AND STORE NEW UNIT# FROM C
F5D3	FEFF	1266	CP	255	;TEST IF NO DRIVE HAS BEEN SELCTD
F5D5	2806	1267	JR	Z,SEL3-\$; YET & SKIP NEXT SEGMENT IF SO
F5D7	23	1268	INC	HL	;POINT TO HEAD POSITION TABLE
F5D8	85	1269	ADD	A,L	; AND ADD IN NEW UNIT# AS INDEX
F5D9	6F	1270	LD	L,A	
F5DA	DB11	1271	IN	A,(TRKREG)	;GET CURRENT HEAD POSITION
F5DC	77	1272	LD	(HL),A	; AND STORE IN TABLE @ HL
F5DD	2160FF	1273	SEL3: LD	HL,TRKTAB	
F5E0	7D	1274	LD	A,L	
F5E1	81	1275	ADD	A,C	;INDEX INTO TABLE TO GET
F5E2	6F	1276	LD	L,A	; HEAD POSITION OF NEW DRIVE
F5E3	7E	1277	LD	A,(HL)	
F5E4	FEFF	1278	CP	255	;TEST IF NEW DRIVE HAS EVER BEEN
F5E6	2804	1279	JR	Z,HOME-\$; SELECTED AND DO A HOME IF NOT
F5E8	D311	1280	OUT	(TRKREG),A	;OUTPUT THE DRIVE'S CURRENT HEAD
F5EA	AF	1281	XOR	A	; POSITION TO THE TRACK REGISTER
F5EB	C9	1282	RET		
		1283 ;			
		1284 ;			
		1285 ;			
F5EC	CDCDF6	1286	HOME: CALL	READY	;CLEAR DISK CONTROLLER
F5EF	C8	1287	RET	Z	;EXIT IF DRIVE NOT READY
F5F0	AF	1288	XOR	A	
F5F1	326AFF	1289	LD	(TRACK),A	;SET TRACK# IN MEM TO ZERO
F5F4	060C	1290	RESTOR: LD	B,RSTCMD	;LOAD B WITH A RESTORE COMMAND
F5F6	CDA2F6	1291	CALL	STEP	;EXECUTE HEAD MOVING OPERATION
F5F9	EE04	1292	XOR	00000100B	;GET TRUE TRACK 0 STATUS
F5FB	E69C	1293	AND	10011100B	;MASK TO ERROR BITS
F5FD	C9	1294	RET		;RETURN 1771 STATUS IN A
		1295 ;			
		1296 ;			
		1297 ;			
F5FE	CDCDF6	1298	SEEK: CALL	READY	;CLEAR DISK CONTROLLER
F601	C8	1299	RET	Z	;EXIT IF DRIVE NOT READY
F602	064D	1300	LD	B,77	;SET TRACKS+1 FOR 8 INCH
F604	DB1C	1301	IN	A,(SYSPIO)	;READ HRDWRE PORT FOR DRIVE TYPE
F606	CB67	1302	BIT	4,A	;BIT 4 SET IF SA800 DRIVES
F608	2002	1303	JR	NZ,EIGHT-\$;IF 8 IN. DRIVES JUMP
F60A	0628	1304	LD	B,40	;DEFAULT SA400,SA450
F60C	79	1305	EIGHT: LD	A,C	;GET TRACK# DATA FROM C
F60D	B8	1306	CP	B	;CHECK FOR MAXIMUM VALID#
F60E	D0	1307	RET	NC	;FORGET IT IF TRACK# LIMIT
F60F	326AFF	1308	LD	(TRACK),A	;STORE TRACK# FOR SEEK
F612	D313	1309	OUT	(DATREG),A	;OUTPUT TRACK # TO 1771
F614	061C	1310	LD	B,SKCMD	;LOAD B WITH A SEEK COMMAND AND
F616	CDA2F6	1311	CALL	STEP	; GO SEEK WITH PROPER STEP RATE
F619	E698	1312	AND	10011000B	;MASK TO READY,SEEK AND CRC ERROR
F61B	C8	1313	RET	Z	; BITS AND RETURN IF ALL GOOD
		1314			
F61C	CDF4F5	1315	CALL	RESTOR	;ELSE TRY TO RE-CALIBRATE HEAD
F61F	C0	1316	RET	NZ	;ERROR IF WE CAN'T FIND TRACK 0
F620	79	1317	LD	A,C	
F621	D313	1318	OUT	(DATREG),A	;OUTPUT TRACK# TO 1771
F623	061C	1319	LD	B,SKCMD	
F625	CDA2F6	1320	CALL	STEP	;TRY TO SEEK THE TRACK AGAIN

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

F628	E698	1321	AND	10011000B	
F62A	C9	1322	RET		;RETURN FINAL SEEK STATUS IN A
		1323 ;			
		1324 ;			
		1325 ;			
F62B	CDCDF6	1326	WRITE: CALL	READY	;CLEAR THE DISK CONTROLLER
F62E	C8	1327	RET	Z	;EXIT IF DRIVE NOT READY
F62F	CDBAF6	1328	CALL	FORCE	
F632	CB77	1329	BIT	6,A	
F634	C0	1330	RET	NZ	;EXIT IF DISK IS WRITE-PROTECTED
F635	06A8	1331	LD	B,WRTCMD	
F637	1806	1332	JR	RDWRT-\$	
		1333			
F639	CDCDF6	1334	READ: CALL	READY	;CLEAR DISK CONTROLLER
F63C	C8	1335	RET	Z	;EXIT IF DRIVE NOT READY
F63D	0688	1336	LD	B,RDCMD	
F63F	226EFF	1337	RDWRT: LD	(IOPTR),HL	;STORE DISK I/O DATA POINTER
F642	216BFF	1338	LD	HL,SECTOR	
F645	71	1339	LD	(HL),C	;STORE SECTOR# FOR READ/WRITE
F646	23	1340	INC	HL	
F647	70	1341	LD	(HL),B	;SAVE READ/WRITE COMMAND BYTE
F648	23	1342	INC	HL	
F649	360B	1343	LD	(HL),RECN	;SET DISK OPERATION RE-TRY COUNT
F64B	F3	1344	RW1: DI		;NO INTERRUPTS DURING DISK I/O
F64C	216600	1345	LD	HL,NMIVEC	;SAVE BYTE AT NMI VECTOR LOCATION
F64F	56	1346	LD	D,(HL)	; IN D FOR DURATION OF READ/WRITE
F650	36C9	1347	LD	(HL),RET	; LOOP AND REPLACE IT WITH A RET
F652	2168FF	1348	LD	HL,RECLN	
F655	46	1349	LD	B,(HL)	;B=NUMBER OF BYTES/SECTOR
F656	0E13	1350	LD	C,DATREG	;C=1771 DATA REGISTER PORT#
F658	2A6EFF	1351	LD	HL,(IOPTR)	;HL=DISK READ/WRITE DATA POINTER
F65B	3A6BFF	1352	LD	A,(SECTOR)	;GET SECTOR NUMBER
F65E	D312	1353	OUT	(SECREG),A	;OUTPUT SECTOR# TO 1771
F660	CDBAF6	1354	CALL	FORCE	;ISSUE A FORCE INTERRUPT COMMAND
F663	CB6F	1355	BIT	5,A	; TO TEST CURRNT HEAD LOAD STATUS
F665	3A6CFF	1356	LD	A,(CMDTYP)	;GET READ OR WRITE COMMAND BYTE
F668	2002	1357	JR	NZ,RW2-\$;JUMP IF HEAD IS ALREADY LOADED
F66A	F604	1358	OR	HLOAD	; ELSE MERGE IN HLD BIT
F66C	CDB2F6	1359	RW2: CALL	CMDOUT	;START THE 1771 DOING IT'S THING
F66F	CB6F	1360	BIT	5,A	;TEST IF COMMAND IS A READ OR WRT
F671	200D	1361	JR	NZ,WLOOP-\$; AND JUMP TO THE CORRECT LOOP
F673	76	1362	RLOOP: HALT		
F674	EDA2	1363	INI		
F676	C273F6	1364	JP	NZ,RLOOP	
F679	CDABF6	1365	CALL	BUSY	;LOOP UNTIL 1771 COMES UN-BUSY
F67C	E69C	1366	AND	10011100B	;MASK OFF TO RDY, NOT FOUND, CRC
F67E	180B	1367	JR	RW3-\$; AND LOST DATA STATUS BITS
		1368			
F680	76	1369	WLOOP: HALT		
F681	EDA3	1370	OUTI		
F683	C280F6	1371	JP	NZ,WLOOP	
F686	CDABF6	1372	CALL	BUSY	
F689	E6BC	1373	AND	10111100B	;MASK OFF AS ABOVE + WRITE FAULT
F68B	216600	1374	RW3: LD	HL,NMIVEC	
F68E	72	1375	LD	(HL),D	;RESTORE BYTE @ NMI VECTOR
F68F	FB	1376	EI		
F690	C8	1377	RET	Z	;RETURN IF NO DISK I/O ERRORS
F691	216DFF	1378	LD	HL,RETRY	
F694	35	1379	DEC	(HL)	;DECREMENT RE-TRY COUNT AND
F695	2002	1380	JR	NZ,RW4-\$; EXECUTE COMAND AGAIN IF NOT=0
F697	B7	1381	OR	A	
F698	C9	1382	RET		;ELSE RETURN 1771 ERROR STATUS
		1383			
F699	216AFF	1384	RW4: LD	HL,TRACK	
F69C	4E	1385	LD	C,(HL)	;GET TRACK# FOR CURRENT OPERATION
F69D	CDFEF5	1386	CALL	SEEK	;TRY TO RE-CALIBRATE THE HEAD
F6A0	18A9	1387	JR	RW1-\$; BEFORE READING OR WRITING AGAIN

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

1388 ;
1389 ;
1390 ;
F6A2 3A67FF 1391 STEP: LD A,(SPEED) ;GET STEP SPEED VARIABLE
F6A5 E603 1392 AND 00000011B
F6A7 B0 1393 OR B ;MRGE WITH SEEK/HOME COMMAND IN B
F6A8 CDB2F6 1394 CALL CMDOUT ;OUTPUT COMMAND AND DELAY
F6AB DB10 1395 BUSY: IN A,(STSREG)
F6AD CB47 1396 BIT 0,A ;TEST BUSY BIT FROM
F6AF 20FA 1397 JR NZ,BUSY-$ ; 1771 AND LOOP TILL=0
F6B1 C9 1398 RET
1399 ;
1400 ;
1401 ;
F6B2 D310 1402 CMDOUT: OUT (CMDREG),A ;OUTPUT A COMMAND TO THE 1771
F6B4 CDB7F6 1403 CALL PAUSE ;WASTE 44 MICROSECONDS
F6B7 E3 1404 PAUSE: EX (SP),HL
F6B8 E3 1405 EX (SP),HL
F6B9 C9 1406 RET
1407 ;
1408 ;
1409 ;
F6BA 3ED0 1410 FORCE: LD A,FINCMD
F6BC CDB2F6 1411 CALL CMDOUT ;ISSUE A FORCE INTERRUPT COMMAND
F6BF DB10 1412 IN A,(STSREG)
F6C1 C9 1413 RET ;RETURN 1771 STATUS REGISTER BITS
1414 ;
1415 ;
1416 ;
F6C2 3E0F 1417 RESTMR: LD A,15
F6C4 3269FF 1418 LD (MOTOR),A ;RE-LOAD MOTOR TURN OFF TIMER
F6C7 CDCCF6 1419 CALL RES2
F6CA DB1C 1420 IN A,(BITDAT) ;GET STATUS OF SYSTEM PIO
F6CC C9 1421 RES2: RET
1422 ;
1423 ;
1424 ;
F6CD CDC2F6 1425 READY: CALL RESTMR ;RESET MOTOR TIMER
F6D0 E607 1426 AND 00000111B ;TEST IF MOTORS HAVE BEEN STOPPED
F6D2 C0 1427 RET NZ ;AND EXIT IF STILL TURNED ON
F6D3 DB1C 1428 IN A,(BITDAT) ;READ THE SYSTEM PORT
F6D5 E5 1429 PUSH HL ;SAVE HL
F6D6 215FFF 1430 LD HL,UNIT ;GET THE DRIVE TO BE SELECTED
F6D9 B6 1431 OR (HL) ;UPDATE THE A REGISTER
F6DA E1 1432 POP HL ;RESTORE HL
1433 ;
1434 ;
1435 ; TURN ON THE SELECTED DRIVE MOTOR AND START TIMING
1436 ; THE ROTATIONAL SPEED TO DETERMINE IF THE DRIVE IS READY
1437 ;
F6DB E5 1438 TURNON: PUSH HL
F6DC C5 1439 PUSH BC
F6DD D31C 1440 OUT (BITDAT),A
F6DF 3E87 1441 LD A,10000111B ;PROGRAM CTC1 FOR TIMER MODE
F6E1 D319 1442 OUT (CTC1),A
F6E3 3E9C 1443 LD A,156 ;INTERRUPT 1000 TIMES/SECOND
F6E5 D319 1444 OUT (CTC1),A
F6E7 21D007 1445 LD HL,2000 ;RESET INDEX PULSE TIMER FOR MAX
F6EA 2270FF 1446 LD (INDTMR),HL ; ALLOWABLE SPIN-UP TIME
1447 ;
F6ED CDBAF6 1448 CALL FORCE ;GET 1771 STATUS BITS AND MASK TO
F6F0 E602 1449 AND 00000010B ; INDEX DETECT BIT
F6F2 47 1450 LD B,A ;SAVE CURRENT STATE OF BIT IN B
F6F3 CD2DF7 1451 TURN2: CALL EDGE ;WAIT FOR THE FIRST CHNG IN INDEX
F6F6 3822 1452 JR C,TURN4-$ ;ABORT IF DRIVE NOT READY
F6F8 2A70FF 1453 TURN3: LD HL,(INDTMR) ; ELSE GET CURRENT TIMER VALUE
F6FB CD2DF7 1454 CALL EDGE

```

```

F6FE 381A      1455      JR      C,TURN4-$
F700 CD2DF7    1456      CALL     EDGE
F703 3815      1457      JR      C,TURN4-$
F705 ED5B70FF  1458      LD      DE,(INDTMR)      ;GET TIMER VALU AT END OF REVOLUTN
F709 ED52      1459      SBC     HL,DE              ;CALCULATE PERIOD OF REVOLUTION
F70B 2272FF    1460      LD      (PERIOD),HL
F70E 11D200    1461      LD      DE,210
F711 B7        1462      OR      A
F712 ED52      1463      SBC     HL,DE              ;TEST IF PERIOD IS TOO LONG AND
F714 30E2      1464      JR      NC,TURN3-$      ; TIME ANOTHER REVOLUTION IF TOO
F716 1E80      1465      LD      E,10000000B
F718 1808      1466      JR      TURNX-$      ;EXIT WITH DRIVE READY INDICATED
                        1467
F71A DB1C      1468 TURN4: IN      A,(BITDAT)      ;TURN THE MOTOR BACK OFF
F71C E6F8      1469      AND     11111000B
F71E D31C      1470      OUT     (BITDAT),A
F720 1E00      1471      LD      E,00000000B      ;INDICATE DRIVE-NOT-READY ERROR
F722 3E03      1472 TURNX: LD      A,00000011B
F724 F3        1473      DI
                        1474      OUT     (CTC1),A      ;KILL INTERRUPT FROM CTC CHNL 2
F725 D319      1474      OUT
F727 FB        1475      EI
F728 C1        1476      POP     BC
F729 E1        1477      POP     HL      ;RESTORE HL AND BC
F72A 7B        1478      LD      A,E
F72B B7        1479      OR      A      ;RETURN DRIVE READY STATUS IN A
F72C C9        1480      RET
                        1481 ;
                        1482 ;
                        1483 ;
F72D CDBAF6    1484 EDGE: CALL     FORCE      ;GET CURRENT INDEX DETECT STATE
F730 E602      1485      AND     00000010B
F732 A8        1486      XOR     B      ;COMPARE TO OLD STATE IN B
F733 2009      1487      JR      NZ,EDGE2-$      ; AND JUMP IF IT HAS CHANGED
F735 3A71FF    1488      LD      A,(INDTMR+1)
F738 CB7F      1489      BIT     7,A      ;ELSE TEST IF INDEX TIMER HAS
F73A 28F1      1490      JR      Z,EDGE-$      ; ROLLED OVER & LOOP AGAIN IF NOT
F73C 37        1491      SCF
F73D C9        1492      RET      ;RETURN CARRY=1 IF TIMEOUT
                        1493
F73E 78        1494 EDGE2: LD      A,B
F73F EE02      1495      XOR     00000010B      ;COMPLIMENT THE INDEX STATE IN B
F741 47        1496      LD      B,A
F742 C9        1497      RET      ;RETURN WITH CARRY=0
                        1498 ;
                        1499 ;
0753          1500 RIEND: EQU     $-ROM      ;SHOULD BE LESS THAN 2K
                        1501 ;
                        1502 ;
                        1503 ;
                        1504 ;
F743 0000      1505 ROMEND: DEFW    0      ;TAIL OF FREE MEMORY LINKED LIST
                        1506 ;
FF00          1507      ORG      RAM
                        1508      INCLUDE MEMORY.ASM
                        1509 ;*****
                        1510 ;*
                        1511 ;*      STORAGE ALLOCATION FOR 256 BYTE SCRATCH RAM      *
                        1512 ;*
                        1513 ;*****
                        1514 ;
                        1515 ;
                        1516
FF00          1517 VECTAB EQU     $      ;INTERRUPT VECTOR TBL STARTS HERE
FF00          1518 SIOVEC: DEFS    16      ;SPACE FOR 8 VECTORS FOR SIO
FF10          1519 CTCVEC: DEFS    8      ;SPACE FOR 4 VECTORS FOR CTC
FF18          1520 SYSVEC: DEFS    4      ;SPACE FOR 2 VECTORS FOR SYS PIO
FF1C          1521 GENVEC: DEFS    4      ;SPACE FOR 2 VECTORS FOR GEN PIO

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

```

1522 ;
1523 ;
1524 ;      KEYBOARD DATA INPUT FIFO VARIABLES
1525
FF20 1526 FIFO:  DEFS    16      ;CONSOLE INPUT FIFO
FF30 1527 FIFCNT: DEFS    1      ;FIFO DATA COUNTER
FF31 1528 FIFIN:  DEFS    1      ;FIFI INPUT POINTER
FF32 1529 FIFOUT: DEFS    1      ;FIFO OUTPUT POINTER
FF33 1530 LOCK:   DEFS    2      ;SHIFT LOCK CHARACTER+FLAG BYTE
1531 ;
1532 ;
1533 ;      STACK POINTER SAVE AND LOCAL STACK FOR INTERRUPT ROUTINES
1534
FF35 1535 SPSAVE: DEFS    2      ;USER STACK POINTER SAVE AREA
FF37 1536 TMPSTK: DEFS   32      ;LOCAL STACK FOR INTERRUPTS
1537 ;
1538 ;
1539 ;      CLOCK-TIMER INTERRUPT VARIABLES
1540
FF57 1541 TIKCNT: DEFS    2      ;BINARY CLOCK TICK COUNTER
FF59 1542 DAY:   DEFS    1      ;CALENDAR DAY
FF5A 1543 MONTH: DEFS    1      ;      MONTH
FF5B 1544 YEAR:   DEFS    1      ;      YEAR
FF5C 1545 HRS:    DEFS    1      ;CLOCK HOURS REGISTER
FF5D 1546 MINS:   DEFS    1      ;      MINUTES RETISTER
FF5E 1547 SECS:   DEFS    1      ;      SECONDS REGISTER
1548 ;
1549 ;
1550 ;      DISK I/O DRIVER VARIABLES
1551
FF5F 1552 UNIT:   DEFS    1      ;CURRENTLY SELECTED DISK#
FF60 1553 TRKTAB: DEFS    7      ;4 DRIVE HEAD POSITION TABLE
FF67 1554 SPEED:  DEFS    1      ;SEEK SPEED FOR 1771 COMMANDS
FF68 1555 RECLen: DEFS    1      ;SECTOR RECORD LENGTH VARIABLE
FF69 1556 MOTOR:  DEFS    1      ;DRIVE MOTOR TURN-OFF TIMER
FF6A 1557 TRACK:  DEFS    1
FF6B 1558 SECTOR: DEFS    1
FF6C 1559 CMDTYP: DEFS    1      ;COMMAND BYTE FOR READS/Writes
FF6D 1560 RETRY:  DEFS    1      ;DISK OPERATION RE-TRY COUNT
FF6E 1561 IOPTR:  DEFS    2      ;DISK I/O BUFFER POINTER
FF70 1562 IND TMR: DEFS    2      ;INDEX HOLE CYCLE PERIOD
FF72 1563 PERIOD: DEFS    2      ;PERIOD OF REVOLUTION OF DISK
1564 ;
1565 ;
1566 ;
1567 ;      CRT OUTPUT DRIVER VARIABLES
1568
FF74 1569 CURSOR: DEFS    2      ;CURSOR POINTER
FF76 1570 CHR SAV: DEFS    1      ;CHARACTER OVERLAYED BY CURSOR
FF77 1571 CSRCHR: DEFS    1      ;CHARACTER USED FOR A CURSOR
FF78 1572 BASE:   DEFS    1      ;CURRENT CONTENTS OF SCROLL REG
FF79 1573 LEADIN: DEFS    1      ;STATE OF LEAD-IN SEQ HANDLER
1574 ;
1575 ;
1576 ;
1577 ;      LISTHEAD POINTER FOR DYNAMIC MEMORY ALLOCATION SCHEME
1578
FF7A 1579 FREPTR: DEFS    2
1580 ;
1581 ;
1582 ;      CONSOLE MONITOR PROGRAM VARIABLES
1583
FF7C 1584 PARAM1: DEFS    2      ;STORAGE FOR NUMBERS READ
FF7E 1585 PARAM2: DEFS    2      ; FROM LINE INPUT BUFFER
FF80 1586 PARAM3: DEFS    2      ; BY 'PARAMS' SUBROUTINE
FF82 1587 PARAM4: DEFS    2
FF84 1588 ESCFLG: DEFS    1      ;CONSOLE ESCAPE FLAG

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U64)

FF85	1589	LAST:	DEFS	2	;LAST ADDRESS USED BY 'MEMDMP'
FF87	1590	LINBUF:	DEFS	80	;CONSOLE LINE INPUT BUFFER
FFD7	1591	RAMEND:	DEFS	1	;END OF SCRATCH RAM
	1592	;			
	1593	;			
	1594				
	1595	;			
	1596			END	

820 MONITOR ROM 2.0

```

0001 ;*****
0002 ;*
0003 ;*          XEROX      820          MONITOR   ROM
0004 ;*
0005 ;*          .. VERSION   2.0
0006 ;*
0007 ;*****
0008 ;
0009 ;
0010 ;          PSECT      ABS
F7F0 0011 ROM      EQU      0F7F0H          ;START OF 4K ROM-TRANSFER CODE
F000 0012 ROM1    EQU      0F000H
F02A 0013 ROM1SP  EQU      ROM1+42          ;PRINT BOARD FOR ROM 1
0014 ;
0015 ;EQUATES FOR ROUTINE CALL IN ROM 2 TO ROM 1
0016 ;
F02A 0017 DUMP    EQU      ROM1SP          ;MEMORY DUMP ROUTINE
F02D 0018 PUT4HS  EQU      DUMP+3          ;DISPLAY ADDRESS IN HL
F030 0019 PUT2HS  EQU      PUT4HS+3        ;DISPLAY DATA
F033 0020 SPACE   EQU      PUT2HS+3        ;DISPLAY SPACE
F036 0021 OUTPUT  EQU      SPACE+3         ;DISPLAY CHARACTER IN A
F039 0022 CRLFS   EQU      OUTPUT+3        ;DISPLAY CRLF
F03C 0023 ECHO     EQU      CRLFS+3        ;DISPLAY CRLF
F03F 0024 ASCHEX   EQU      ECHO+3         ;CONVERT ASCII TO HEX
F042 0025 PNEXT   EQU      ASCHEX+3        ;DISPLAY MESSAGE
0026 ;
0027 ;
0028 ;
FF00 0029 RAM     EQU      0FF00H          ;START OF 256 BYTE RAM
3000 0030 CRTMEM  EQU      3000H           ;BASE OF 4K CRT MEMORY
0031 ;
0004 0032 EOT      EQU      04H
000D 0033 CR       EQU      0DH
0034 ;
F7F0 0035         ORG      ROM
0036 ;
0037 ;
0038 ;SPRING BOARD FOR ROM 1
0039 ;
F7F0 C308F8 0040         JP      MEMDMP          ;MEMORY DUMP IN HEX AND ASCII
F7F3 C3C6F8 0041         JP      BLOCK           ;BLOCK MOVE
F7F6 C32AF8 0042         JP      VIEW            ;MEMORY EXAM AND CHANGE
F7F9 C3B8F8 0043         JP      FILL            ;MEMORY FILL
F7FC C36AF8 0044         JP      TEST            ;RAM DIAGNOSTICS
F7FF C35DF8 0045         JP      GOTO            ;PROGRAM EXECUTION
F802 C3DEF8 0046         JP      VERCMD          ;MEMORY COMPARE
F805 C3F4F8 0047         JP      TYPE            ;TYPEWRITER MODE
0048 ;
0049 ;
0050 ;
0051 ;
0052 ;          INCLUDE MON2.ASM
0053 ;*****
0054 ;*
0055 ;*          BASIC HEX MONITOR FOR Z-80 PROCESSORS
0056 ;*
0057 ;*****
0058 ;
ADDR  CODE  STMT SOURCE STATEMENT  Z-80  ASSEMBLER          PAGE 0002

0059 ;
0060 ;
0061 ;
0062 ;
0063 ;

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U63)

```

0064 ;      -- MEMORY DUMP COMMAND --
0065 ;
F808 3D      0066 MEMDMP: DEC    A      ;CHECK PARAMETER COUNT
F809 2806     0067          JR      Z,MDMP2-$
F80B 3D      0068          DEC    A
F80C 2808     0069          JR      Z,MDMP3-$
F80E 2A85FF   0070 MDMP1: LD      HL,(LAST)
F811 111000   0071 MDMP2: LD      DE,16
F814 180D     0072          JR      MDMP3B-$
          0073
F816 EB      0074 MDMP3: EX      DE,HL
F817 ED52     0075          SBC    HL,DE      ;DERIVE BYTECNT FOR DUMP RANGE
F819 0604     0076          LD      B,4
F81B CB3C     0077 MDMP3A: SRL    H      ;DIVIDE BYTECOUNT BY 16
F81D CB1D     0078          RR      L
F81F 10FA     0079          DJNZ   MDMP3A-$
F821 23       0080          INC    HL
F822 EB      0081          EX      DE,HL
F823 CD2AF0   0082 MDMP3B: CALL   DUMP      ;DUMP DE*16 BYTES STRTING AT HL
F826 2285FF   0083          LD      (LAST),HL
F829 C9       0084          RET
          0085 ;
          0086 ;
          0087 ;
          0088 ;
          0089 ;
0090 ;      -- MEMORY EXAMINE COMMAND --
0091 ;
F82A CDADF8   0092 VIEW:  CALL    MDATA
F82D CD3CF0   0093          CALL   ECHO
F830 FE0D     0094          CP      CR
F832 2824     0095          JR      Z,VIEW4-$
F834 FE2D     0096          CP      '-'
F836 2822     0097          JR      Z,VIEW5-$
F838 FE2C     0098          CP      ','
F83A 2005     0099          JR      NZ,VIEW2-$
F83C CD3CF0   0100          CALL   ECHO
F83F 1813     0101          JR      VIEW3-$
          0102
F841 CD3FF0   0103 VIEW2:  CALL    ASCHEX
F844 3F       0104          CCF
F845 D0       0105          RET      NC
F846 07       0106          RLCA
F847 07       0107          RLCA
F848 07       0108          RLCA
F849 07       0109          RLCA
F84A 4F       0110          LD      C,A
F84B CD3CF0   0111          CALL   ECHO
F84E CD3FF0   0112          CALL   ASCHEX
F851 3F       0113          CCF
F852 D0       0114          RET      NC
F853 B1       0115          OR      C
F854 77       0116 VIEW3:  LD      (HL),A
F855 CD97F8   0117          CALL   CHECK
F858 23       0118 VIEW4:  INC     HL
F859 23       0119          INC     HL
F85A 2B       0120 VIEW5:  DEC     HL
F85B 18CD     0121          JR      VIEW-$
          0122 ;
          0123 ;
          0124 ;
          0125 ;      -- JUMP TO MEMORY LOCATION COMMAND --
          0126 ;
F85D 3D      0127 GOTO:  DEC     A      ;CHECK PARAMETER COUNT
F85E 37       0128          SCF
F85F C0       0129          RET      NZ
F860 E5       0130          PUSH   HL

```

```

F861 DDE1      0131      POP      IX
F863 CD68F8    0132      CALL     CALLX      ;CALL ADDRESS PASSED IN HL
F866 B7        0133      OR       A
F867 C9        0134      RET
                                ;RETURN IF WE GET BACK AGAIN
                                0135 ;
F868 DDE9      0136 CALLX: JP      (IX)      ;JUMP TO ADDRESS IN IX
                                0137 ;
                                0138 ;
                                0139 ;      -- MEMORY READ/WRITE DIAGNOSTIC COMMAND --
                                0140 ;
F86A FE02      0141 TEST:  CP      2          ;CHECK PARAMETER COUNT
F86C 37        0142      SCF
F86D C0        0143      RET      NZ
F86E 13        0144      INC      DE
F86F 5A        0145      LD       E,D        ;GET ENDING PAGE ADDRESS INTO E
F870 54        0146      LD       D,H        ;GET STARTING PAGE ADDR INTO D
F871 0600      0147      LD       B,0        ;INITIALIZE PASS COUNTER
F873 62        0148 TEST1: LD       H,D        ;POINT HL TO START OF BLOCK
F874 2E00      0149      LD       L,0
F876 7D        0150 TEST2: LD       A,L
F877 AC        0151      XOR      H          ;GENERATE TEST BYTE
F878 A8        0152      XOR      B
F879 77        0153      LD       (HL),A      ;STORE BYTE IN RAM
F87A 23        0154      INC      HL
F87B 7C        0155      LD       A,H
F87C BB        0156      CP       E          ;CHECK FOR END OF TEST BLOCK
F87D 20F7      0157      JR       NZ,TEST2-$
                                0158 ;
                                0159      LD       H,D        NOW READ BACK EACH BYTE & COMPARE
F87F 62        0160      LD       L,0        ;POINT HL BACK TO START
F880 2E00      0161 TEST3: LD       A,L
F882 7D        0162      LD       A,L
F883 AC        0163      XOR      H          ;RE-GENERATE TEST BYTE DATA
F884 A8        0164      XOR      B
F885 CD97F8    0165      CALL     CHECK      ;VERIFY MEMORY DATA STILL GOOD
F888 C0        0166      RET      NZ        ;EXIT IF ESCAPE REQ IS INDICATED
F889 23        0167      INC      HL        ; ELSE GO ON TO NEXT BYTE
F88A 7C        0168      LD       A,H
F88B BB        0169      CP       E          ;CHECK FOR END OF BLOCK
F88C 20F4      0170      JR       NZ,TEST3-$
F88E 04        0171      INC      B          ;BUMP PASS COUNT
F88F 3E2B      0172      LD       A,'+'
F891 CD36F0    0173      CALL     OUTPUT      ;PRINT '+' AND ALLOW FOR EXIT
F894 28DD      0174      JR       Z,TEST1-$   ;DO ANOTHER PASS IF NO ESCAPE
F896 C9        0175 ;
                                0176 ;
                                0177 ;
F897 BE        0178 CHECK: CP      (HL)
F898 C8        0179      RET      Z          ;RETURN IF (HL)=A
F899 F5        0180      PUSH     AF
F89A CDADF8    0181      CALL     MDATA      ;PRINT WHAT WAS ACTUALLY READ
F89D CD42F0    0182      CALL     PNEXT
F8A0 73686F75 0183      DEFB     'should='
                                6C643D
F8A7 04        0184      DEFB     EOT
F8A8 F1        0185      POP      AF
F8A9 CD30F0    0186      CALL     PUT2HS      ;PRINT WHAT SHD HAVE BEEN READ
F8AC C9        0187      RET
                                0188 ;
                                0189 ;
F8AD CD39F0    0190 MDATA: CALL     CRLFS
F8B0 CD2DF0    0191      CALL     PUT4HS
F8B3 7E        0192      LD       A,(HL)
F8B4 CD30F0    0193      CALL     PUT2HS
F8B7 C9        0194      RET
                                0195 ;
                                0196 ;

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U63)

```

0197 ;
0198 ;      -- FILL MEMORY WITH CONSTANT COMMAND --
0199 ;
F8B8 FE03 0200 FILL: CP      3      ;CHECK IF PARAMETER COUNT=3
F8BA 37    0201      SCF
F8BB C0    0202      RET      NZ
F8BC 71    0203 FILL1: LD      (HL),C
F8BD E5    0204      PUSH    HL
F8BE B7    0205      OR      A
F8BF ED52  0206      SBC     HL,DE      ;COMPARE HL TO END ADDRESS IN DE
F8C1 E1    0207      POP     HL
F8C2 23    0208      INC     HL      ;ADVANCE POINTER AFTER COMPARISON
F8C3 38F7  0209      JR      C,FILL1-$
F8C5 C9    0210      RET
0211 ;
0212 ;
0213 ;
0214 ;
0215 ;      -- MEMORY BLOCK MOVE COMMAND --
0216 ;
F8C6 FE03 0217 BLOCK: CP      3      ;CHECK IF PARAMETER COUNT=3
F8C8 37    0218      SCF
F8C9 C0    0219      RET      NZ
F8CA CDD3F8 0220      CALL    BLOCAD
F8CD 79    0221      LD      A,C
F8CE B0    0222      OR      B
F8CF C8    0223      RET      Z      ;EXIT NOW IF BC=0
F8D0 EDB0  0224      LDIR
F8D2 C9    0225      RET
0226 ;
0227 ;
0228 ;
F8D3 EB    0229 BLOCAD: EX     DE,HL
F8D4 B7    0230      OR      A      ;CLEAR CARRY
F8D5 ED52  0231      SBC     HL,DE      ;GET DIFFERENCE BETWEEN
F8D7 EB    0232      EX     DE,HL      ;HL & DE FOR BYTECOUNT
F8D8 D5    0233      PUSH    DE
F8D9 C5    0234      PUSH    BC
F8DA D1    0235      POP     DE      ;GET OLD BC INTO DE
F8DB C1    0236      POP     BC
F8DC 03    0237      INC     BC      ;GET COUNT+1 INTO BC
F8DD C9    0238      RET
0239 ;
0240 ;
0241 ;
0242 ;      -- MEMORY BLOCK COMPARE COMMAND --
0243 ;
F8DE FE03 0244 VERCMD: CP      3      ;CHECK IF PARAMETER COUNT=3
F8E0 37    0245      SCF
F8E1 C0    0246      RET      NZ
F8E2 CDD3F8 0247      CALL    BLOCAD
F8E5 1808  0248      JR      VERF2-$
0249
F8E7 1A    0250 VERF1: LD      A,(DE)
F8E8 CD97F8 0251      CALL    CHECK      ;COMPARE DATA @ DE AND @ HL
F8EB C0    0252      RET      NZ      ;EXIT IF ESCAPE REQ IS INDICATED
F8EC 23    0253      INC     HL
F8ED 13    0254      INC     DE
F8EE 0B    0255      DEC     BC
F8EF 78    0256 VERF2: LD      A,B
F8F0 B1    0257      OR      C
F8F1 20F4  0258      JR      NZ,VERF1-$
F8F3 C9    0259      RET
0260 ;
0261 ;
0262 ;
0263 ;

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U63)

```

0264          INCLUDE TYPE.ASM
0265 *****
0266 ;*
0267 ;*
0268 ;*          XEROX 820 TYPEWRITER MODE
0269 ;*
0270 ;*****
0271 ;
F018 0272 SIOOUT      EQU      0F018H ;SIO CH B OUTPUT ROUTINE
F006 0273 CONST      EQU      0F006H ;KEY BOARD STATUS ROUTINE
F009 0274 CONIN      EQU      0F009H ;KEY BOARD DATA ROUTINE
F00C 0275 CRTOUT     EQU      0F00CH ;CRT OUTPUT ROUTINE
F000 0276 COLD       EQU      0F000H ;SOFTWARE RESET
0277 ;
0278 ;
F8F4 0279 TYPE:      ORG      $
0280 ;
0281 ;
0282 ;
0283 ;SET UP PRINTER BAUD RATE
0284 ;
F8F4 7D 0285          LD      A,L          ;GET BAUD RATE IN L
F8F5 E60F 0286        AND      0FH          ;USE VALUES FROM 0 TO 15
F8F7 2002 0287        JR      NZ,BAUD-$      ;DEFLT ZERO FOR 1200 BAUD
F8F9 3E07 0288        LD      A,7
0289 ;
F8FB 0290 BAUD:      ;
0291 ;
F8FB D30C 0292        OUT      (0CH),A        ;SET UP BAUD RATE FOR CH B
F8FD 3E1A 0293        LD      A,01AH        ;CLR SCRN TO CURSOR TO LEFT
F8FF CD0CF0 0294      CALL     CRTOUT
F902 CD42F0 0295      CALL     PNEXT          ;DISPLAY THE FLWNG MSGS
0296 ;
F905 0297 MESS      ;
0298 ;
F905 2E2E2E38 0299      DEFM     '...820 TYPEWRITER VER. 1.0...'
32302054
59504557
52495445
52202056
45522E20
312E302E
2E2E
F923 0D0A 0300      DEFB     0DH,0AH          ;CR,LF
F925 20202050 0301      DEFM     ' PRESS CTRL+X TO EXIT'
52455353
20435452
4C2B5820
544F2045
584954
F93C 0D0A 0302      DEFB     0DH,0AH          ;CR,LF
F93E 04 0303      DEFB     04H          ;END OF TEXT
0304 ;
0305 ;
0306 ;
F93F 211CFA 0307      LD      HL,PRINI        ;GET PRT INIT COMMANDS
F942 0609 0308      LD      B,9          ;GET COMMAND COUNT
F944 CD14FA 0309      CALL     INILUP        ;RESET PRINTER
0310 ;
0311 ;
F947 0E05 0312      LD      C,5          ;SET COUNTER OF 5 SPACES
F949 1619 0313      LD      D,25        ;SET COUNTER FOR 25 TABS
F94B 79 0314      LD      A,C
F94C 0315 TABSET:    ;
0316 ;
0317 ;
F94C 3231FA 0318      LD      (TBCMD+7),A      ;SAVE TAB POSITION

```

F94F	212AFA	0319	LD	HL,TBCMD	;SEND TAB COMMAND TO PRT
F952	060F	0320	LD	B,15	;SEND ABS TAB & SET TAB
F954	CD14FA	0321	CALL	INILUP	
F957	3E05	0322	LD	A,5	;SET UP NEXT TAB POSITN
F959	81	0323	ADD	A,C	
F95A	4F	0324	LD	C,A	;AND SAVE IT
F95B	15	0325	DEC	D	
F95C	20EE	0326	JR	NZ,TABSET-\$;UNTIL 25 TABS ARE SET
		0327 ;			
		0328 ;SEND CR			
		0329 ;			
F95E	3E0D	0330	LD	A,ODH	
F960	CD18F0	0331	CALL	SIOOUT	;AND SEND CR
		0332 ;			
		0333 ;SET UP LEFT MARGIN AT 12			
		0334 ;			
F963	2125FA	0335	LD	HL,LMTAB	;SET UP COMMAND TBL FOR
		0336 ;			;LEFT MARGIN
F966	0605	0337	LD	B,5	;SEND CARRIAGE TO COL 12
F968	CD14FA	0338	CALL	INILUP	;& SET LEFT MARGIN THERE
F96B	3E0C	0339	LD	A,12	;INIT MARGIN AND COL COUNT
F96D	2142FA	0340	LD	HL,LPLC	
F970	77	0341	LD	(HL),A	
F971	23	0342	INC	HL	
F972	77	0343	LD	(HL),A	
F973	AF	0344	XOR	A	
F974	23	0345	INC	HL	
F975	77	0346	LD	(HL),A	;RESET ESCAPE SEQUENCE
		0347 ;			
		0348 ;			
F976		0349 TYPLUP:			
		0350 ;			
F976	CD06F0	0351	CALL	CONST ;KEY IN INPUT BUFFER?	
F979	28FB	0352	JR	Z,TYPLUP-\$;WAIT UNTIL KEY IN INPUT BFR
		0353 ;			
		0354 ;KEY IS AVAILABLE			
		0355 ;			
F97B	3A44FA	0356	LD	A,(ESCKEY)	
F97E	D601	0357	SUB	1	
F980	CE00	0358	ADC	A,0	;DECRS ESC CONTR UNTIL ZERO
F982	3244FA	0359	LD	(ESCKEY),A	
		0360 ;			
		0361 ;			
F985		0362 KEYIN:			
		0363 ;			
F985	CD09F0	0364	CALL	CONIN	;GET KEY IN INPUT BUFFER
		0365 ;	LD	DE,CRTL C	;GET CRT COL COUNTER ADRS
F988	2142FA	0366	LD	HL,LPLC	;GET PRT COL COUNTER ADRS
F98B	4F	0367	LD	C,A	;SAVE KEY IN REGISTER C
F98C	FE20	0368	CP	020H	;PRINTABLE CHARACTER?
F98E	D2F0F9	0369	JP	NC,PRTKEY	;YES PRINTABLE CHARACTER
		0370 ;			
		0371 ;CONTROL KEY			
		0372 ;			
F991		0373 CNTKEY:			
		0374 ;			
		0375 ;			
F991	FE0D	0376	CP	ODH	;KEY IS CR?
F993	200F	0377	JR	NZ,NOCR-\$;NOT A CR
		0378 ;			
		0379 ;GET A CR HERE			
		0380 ;			
F995		0381 CARET:			
		0382 ;			
F995	3A43FA	0383	LD	A,(LFMG)	;GET LEFT MARGIN
F998	77	0384	LD	(HL),A	;SET PRT COL CNT TO LFT MRGN
F999	2139FA	0385	LD	HL,CRLF	;SEND CR AND LF TO PRT

F99C	0609	0386	LD	B,9	
F99E	CD14FA	0387	CALL	INILUP	
F9A1	C376F9	0388	JP	TYPLUP	;AND GET ANOTHER KEY
		0389 ;			
		0390 ;NOT A CR KEY			
		0391 ;			
F9A4		0392 NOCR:			
		0393 ;			
F9A4	FE18	0394	CP	18H	;KEY IS CNTR-X?
F9A6	C2B4F9	0395	JP	NZ,NOX	;NO, TEST FOR OTHER KEY
F9A9	2139FA	0396	LD	HL,CRLF	;SEND CRLF TO PRINTER
F9AC	0609	0397	LD	B,9	
F9AE	CD14FA	0398	CALL	INILUP	
F9B1	C300F0	0399	JP	COLD	
F9B4		0400 NOX:			
		0401 ;			
F9B4	FE1B	0402	CP	01BH	;KEY IS ESC KEY?
F9B6	2008	0403	JR	NZ,NOESC-\$;NOT AN ESCAPE KEY
		0404 ;			
		0405 ;ESCAPE KEY PRESSED			
		0406 ;			
F9B8	3E03	0407	LD	A,3	;SET UP 3 BYTE ESC KEY SEQ
F9BA	3244FA	0408	LD	(ESCKEY),A	
F9BD	C30DFA	0409	JP	PRTOUT	;SEND ESC KEY TO PRT & GET
		0410 ;			;ANOTHER KEY
		0411 ;			
		0412 ;NOT AN ESCAPE KEY			
		0413 ;			
F9C0		0414 NOESC:			
		0415 ;			
F9C0	FE09	0416	CP	09H	;KEY IS TAB KEY?
F9C2	201B	0417	JR	NZ,NOTAB-\$;NOT A TAB KEY
		0418 ;			
		0419 ;TAB KEY PRESSED			
		0420 ;			
		0421 ;COMPARE CURRENT PRT COLUMN POSITION WITH LIST OF TAB COLUMN			
		0422 ;AND USE THE NEXT LARGER VALUE OF TAB POSITION TO BE			
		0423 ;CURRENT POSITION			
		0424 ;			
F9C4	DD2145FA	0425	LD	IX,TABTBL	;SET UP ADDRESS OF TAB TBL
F9C8	46	0426	LD	B,(HL)	;SET UP CURRENT PRT POSITN
		0427 ;			
F9C9		0428 TBLUP:			
		0429 ;			
F9C9	DD7E00	0430	LD	A,(IX)	;GET TAB COLUMN NUMBER
F9CC	A7	0431	AND	A	;TAB COLUMN IS ZERO?
F9CD	280B	0432	JR	Z,COL132-\$;ERROR, TAB NOT FOUND
F9CF	DD23	0433	INC	IX	;GET NXT ADDR OF TAB COL
F9D1	B8	0434	CP	B	;CMPRE WITH CURNT PRT POSITN
F9D2	38F5	0435	JR	C,TBLUP-\$;UNTIL TAB COL NUMBER IS
F9D4	28F3	0436	JR	Z,TBLUP-\$;GREATER
		0437 ;			
F9D6	77	0438	LD	(HL),A	;THEN USE IT AS CURRENT COL
F9D7	C30DFA	0439	JP	PRTOUT	;& SEND TAB KEY OUT TO PRT
		0440 ;			
		0441 ;			
		0442 ;PRINT BELL TO INDICATE AT RIGHT MARGIN ON THE PRINTER			
		0443 ;			
		0444 ;			
		0445 ;			
F9DA		0446 COL132:			
F9DA		0447 COLO:			
		0448 ;			
F9DA	0E07	0449	LD	C,07H	;PRINT BELL
F9DC	C30DFA	0450	JP	PRTOUT	;AND GET ANOTHER KEY
		0451 ;			
		0452 ;			

		0453 ;NOT A TAB KEY		
		0454 ;		
F9DF		0455 NOTAB:	;	
		0456 ;		
F9DF	FE08	0457	CP 08H	;KEY IS BACK SPACE KEY?
F9E1	202A	0458	JR NZ,PRTOUT-\$;NOT A BACK SPACE KEY
		0459 ;		;PRINT KEY WITHOUT COL COUNT
		0460 ;		;INCREAMENT
		0461 ;		
		0462 ;BACK SPACE KEY PRESSED		
		0463 ;		
		0464		
		0465		
F9E3	3A43FA	0466	LD A,(LFMG)	;GET LEFT MARGIN IN B
F9E6	47	0467	LD B,A	
F9E7	7E	0468	LD A,(HL)	;GET PRINTER COLUMN COUNT
F9E8	B8	0469	CP B	;AT LEFT MARGIN?
F9E9	CADAF9	0470	JP Z,COLO	;YES, PRINT BELL
		0471 ;		
		0472 ;		
		0473 ;		
		0474 ;		
F9EC	35	0475	DEC (HL)	;DECREASE PRT COL COUNT
		0476 ;		;BY ONE
F9ED	C30DFA	0477	JP PRTOUT	;PRINT BACK SPACE
		0478 ;		
		0479 ;		
		0480 ;		
		0481 ;		
		0482 ;PRINTABLE CHARACTER		
		0483 ;		
		0484 ;		
F9F0		0485 PRTKEY:	;	
		0486 ;		
F9F0	7E	0487	LD A,(HL)	;GET PRT COLUMN COUNT
F9F1	FE84	0488	CP 132	;REACH RIGHT MARGIN?
F9F3	CADAF9	0489	JP Z,COL132	;YES, PRINT BELL
		0490 ;		
		0491 ;		
		0492 ;		
		0493 ;		
F9F6	3A44FA	0494	LD A,(ESCKEY)	;KEY IS WITHIN ESC SEQ?
F9F9	A7	0495	AND A	
F9FA	280D	0496	JR Z,INCCOL-\$;NO, PRINT CHAR WITH INCRSE
		0497 ;		;COLUMN COUNT
		0498 ;		
F9FC	79	0499	LD A,C	;GET CHARACTER
F9FD	FE39	0500	CP 039H	;CHAR IS NUMBER 9?
F9FF	C20DFA	0501	JP NZ,PRTOUT	;NO,JUST SEND CHAR TO PRT
		0502 ;		
		0503 ;SET NEW LEFT MARGIN		
		0504 ;		
FA02	7E	0505	LD A,(HL)	;GET CURRENT COLUMN COUNT
FA03	3243FA	0506	LD (LFMG),A	;AS LEFT MARGIN
FA06	C30DFA	0507	JP PRTOUT	;SEND CHAR TO PRT
		0508 ;		
FA09		0509 INCCOL:	;	
		0510 ;		
		0511 ;INCREASE COLLUMN COUNTER		
		0512 ;		
FA09	34	0513	INC (HL)	;INCREASE PRT COL COUNTER
		0514 ;		;BY ONE
		0515 ;		
FA0A	C30DFA	0516	JP PRTOUT	;PRINT CHAR & GET ANOTHER KEY
		0517 ;		
		0518 ;		
		0519 ;		

FA0D		0520	PRTOUT:			
FA0D	79	0521		LD	A,C	;GET PRINT CHARACTER
FA0E	CD18F0	0522		CALL	SIOOUT	;SEND IT TO USART PORT B
FA11	C376F9	0523		JP	TYPLUP	;GET ANOTHER KEY
		0524				
		0525				
FA14		0526	INILUP			
		0527				
FA14	7E	0528		LD	A,(HL)	;GET COMMAND
FA15	CD18F0	0529		CALL	SIOOUT	;SEND IT TO SIO PORT B
FA18	23	0530		INC	HL	
FA19	10F9	0531		DJNZ	INILUP-\$;UNTIL B BYTES ARE SENT
FA1B	C9	0532		RET		
		0533				
		0534				
		0535				
		0536				
		0537				*****
		0538				*
		0539				*
		0540				*
		0541				*****
		0542				
		0543				
		0544				
		0545				;PRINTER INITIALIZATION COMMANDS
		0546				;PRINTER RESET COMMAND
		0547				;12 SPACES
		0548				;SET LEFT MARGIN TO COLUMN 12
		0549				
FA1C	1B0D50	0550	PRTINI:	DEFB	01BH,0DH,050H	;ESC CR P SEQUENCE
FA1F	00000000	0551		DEFB	0,0,0,0,0,0	
	0000					
FA25	1B090C	0552	LMTAB:	DEFB	1BH,09H,0CH	;TAB TO COLUMN 12
FA28	1B39	0553		DEFB	1BH,39H	;SET LEFT MARGIN
		0554				
		0555				;SET TAB AT EVERY 5 COLUMN
		0556				
FA2A	00000000	0557	TBCMD:	DEFB	0,0,0,0,0	
	00					
FA2F	1B0900	0558		DEFB	1BH,09H,00	;MOVE CARRIAGE TO COL. XX
FA32	1B31	0559		DEFB	1BH,31H	;SET TAB THERE
FA34	00000000	0560		DEFB	0,0,0,0,0	
	00					
		0561				
		0562				
		0563				
FA39	0D0A	0564	CRLF:	DEFB	0DH,0AH	
FA3B	00000000	0565		DEFB	0,0,0,0,0,0,0	
	000000					
		0566				
		0567				
		0568				
		0569	CRTLC:	DEFB	0	;CRT COLUMN COUNT
FA42	0C	0569	LPLC:	DEFB	12	;PRT COLUMN COUNT
FA43	0C	0570	LFMG:	DEFB	12	;PRT LEFT MARGIN
FA44	00	0571	ESCKEY:	DEFB	0	;NO ESCAPE KEY SEQUENCE
		0572				
		0573				
		0574				
		0575				;TAB POSITION TABLE
		0576				
		0577				
FA45	050A0F14	0578	TABTEL:	DEFB	5,10,15,20,25,30,35,40,45,50	
	191E2328					
	2D32					
FA4F	373C4146	0579		DEFB	55,60,65,70,75,80,85,90,95,100	
	4B50555A					

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U63)

```

5F64
FA59 696E7378 0580          DEFB  105,110,115,120,125,130,135,140,0
      7D82878C
      00
      0581 ;
      0582 ;
      0583 ;
      0584 ;
      0585 ;
FA62 0000 0586 ROMEND: DEFW  0          ;TAIL OF FREE MEMORY LINKED LIST
      0587 ;
      0588          ORG      RAM
      0589          INCLUDE MEMORY.ASM
      0590 ;*****
      0591 ;*
      0592 ;*      STORAGE ALLOCATION FOR 256 BYTE SCRATCH RAM      *
      0593 ;*
      0594 ;*****
      0595 ;
      0596 ;
      0597
FF00 0598 VECTAB EQU    $          ;INTERRUPT VECTOR TBL STARTS HERE
FF00 0599 SIOVEC: DEFS  16          ;SPACE FOR 8 VECTORS FOR SIO
FF10 0600 CTCVEC: DEFS  8          ;SPACE FOR 4 VECTORS FOR CTC
FF18 0601 SYSVEC: DEFS  4          ;SPACE FOR 2 VECTORS FOR SYS PIO
FF1C 0602 GENVEC: DEFS  4          ;SPACE FOR 2 VECTORS FOR GEN PIO
      0603 ;
      0604 ;
      0605 ;      KEYBOARD DATA INPUT FIFO VARIABLES
      0606
FF20 0607 FIFO:  DEFS  16          ;CONSOLE INPUT FIFO
FF30 0608 FIFCNT: DEFS  1          ;FIFO DATA COUNTER
FF31 0609 FIFIN:  DEFS  1          ;FIFI INPUT POINTER
FF32 0610 FIFOUT: DEFS  1          ;FIFO OUTPUT POINTER
FF33 0611 LOCK:   DEFS  2          ;SHIFT LOCK CHARACTER+FLAG BYTE
      0612 ;
      0613 ;
      0614 ;      STACK POINTER SAVE AND LOCAL STACK FOR INTERRUPT ROUTINES
      0615

FF35 0616 SPSAVE: DEFS  2          ;USER STACK POINTER SAVE AREA
FF37 0617 TMPSTK: DEFS  32         ;LOCAL STACK FOR INTERRUPTS
      0618 ;
      0619 ;
      0620 ;      CLOCK-TIMER INTERRUPT VARIABLES
      0621
FF57 0622 TIKCNT: DEFS  2          ;BINARY CLOCK TICK COUNTER
FF59 0623 DAY:   DEFS  1          ;CALENDAR DAY
FF5A 0624 MONTH: DEFS  1          ;      MONTH
FF5B 0625 YEAR:  DEFS  1          ;      YEAR
FF5C 0626 HRS:   DEFS  1          ;CLOCK HOURS REGISTER
FF5D 0627 MINS:  DEFS  1          ;      MINUTES RETISTER
FF5E 0628 SECS:  DEFS  1          ;      SECONDS REGISTER
      0629 ;
      0630 ;
      0631 ;      DISK I/O DRIVER VARIABLES
      0632
FF5F 0633 UNIT:   DEFS  1          ;CURRENTLY SELECTED DISK#
FF60 0634 TRKTAB: DEFS  7          ;4 DRIVE HEAD POSITION TABLE
FF67 0635 SPEED:  DEFS  1          ;SEEK SPEED FOR 1771 COMMANDS
FF68 0636 RECLEN: DEFS  1          ;SECTOR RECORD LENGTH VARIABLE
FF69 0637 MOTOR:  DEFS  1          ;DRIVE MOTOR TURN-OFF TIMER
FF6A 0638 TRACK:  DEFS  1
FF6B 0639 SECTOR: DEFS  1
FF6C 0640 CMDTYP: DEFS  1          ;COMMAND BYTE FOR READS/Writes
FF6D 0641 RETRY:  DEFS  1          ;DISK OPERATION RE-TRY COUNT
FF6E 0642 IOPTR:  DEFS  2          ;DISK I/O BUFFER POINTER

```

ROM LISTINGS
MONITOR ROM VERSION 2.0 (U63)

```

FF70      0643 INDTMR: DEFS      2      ;INDEX HOLE CYCLE PERIOD
FF72      0644 PERIOD: DEFS      2      ;PERIOD OF REVOLUTION OF DISK
          0645 ;
          0646 ;
          0647 ;
          0648 ;      CRT OUTPUT DRIVER VARIABLES
          0649
FF74      0650 CURSOR: DEFS      2      ;CURSOR POINTER
FF76      0651 CHRSAV: DEFS      1      ;CHARACTER OVERLAYED BY CURSOR
FF77      0652 CSRCHR: DEFS      1      ;CHARACTER USED FOR A CURSOR
FF78      0653 BASE:  DEFS      1      ;CURRENT CONTENTS OF SCROLL REG
FF79      0654 LEADIN: DEFS      1      ;STATE OF LEAD-IN SEQ HANDLER
          0655 ;
          0656 ;
          0657 ;
          0658 ;      LISTHEAD POINTER FOR DYNAMIC MEMORY ALLOCATION SCHEME
          0659
FF7A      0660 FREPTR: DEFS      2
          0661 ;
          0662 ;
          0663 ;      CONSOLE MONITOR PROGRAM VARIABLES
          0664
FF7C      0665 PARAM1: DEFS      2      ;STORAGE FOR NUMBERS READ
FF7E      0666 PARAM2: DEFS      2      ; FROM LINE INPUT BUFFER
FF80      0667 PARAM3: DEFS      2      ; BY 'PARAMS' SUBROUTINE
FF82      0668 PARAM4: DEFS      2
FF84      0669 ESCFLG: DEFS      1      ;CONSOLE ESCAPE FLAG
FF85      0670 LAST:  DEFS      2      ;LAST ADDRESS USED BY 'MEMDMP'
FF87      0671 LINBUF: DEFS     80      ;CONSOLE LINE INPUT BUFFER
FFD7      0672 RAMEND: DEFS      1      ;END OF SCRATCH RAM
          0673 ;
          0674 ;
          0675
          0676 ;
          0677      END

```



```

.Z80
.SFCOND
;*****
;*
;*  -- CUSTOM BIOS FOR CP/M VERSION 2.2  --
;*      8-INCH DISK VERSION
;*
;*      APRIL 1981
;*
;*      CBIOS FOR XEROX CP/M DISK
;*      COMBINED VERSION FOR 5.25" AND 8" - JUNE 1981
;*
;*****
;
;
;   ASEG
;
003C      MSIZE      EQU      60          ;MEMORY CAPACITY IN KBYTES
F000      MONITR     EQU      0F000H      ;BASE OF SYSTEM MONITOR

0028      EXTRA     EQU      MSIZE-20
A000      BASE      EQU      EXTRA*1024

D400      CCP        EQU      3400H+BASE  ;CONSOLE COMMAND PROCESSOR
DC06      BDOS       EQU      3C06H+BASE  ;OPERATING SYSTEM ENTRY POINT
EA00      CBIOS      EQU      4A00H+BASE  ;BASE OF CUSTOM BIOS

;
;
;   EQUATES TO SELECT THE CONDITIONAL ASSEMBLY
;   FOR 5.25 OR 8 INCH DISKS
;
;   THE EQUATES DSKTY5 OR DSKTY8 ARE USED FOR
;   CONDITIONAL ASSEMBLY CONTROL
;
;   ONE CONDITIONAL ASSEMBLY FLAG SHOULD BE ON
;   AND THE OTHER SHOULD BE OFF AT ALL TIMES.
;
0001      DSKTY5     EQU      1          ;5.25 INCH DISK TYPE FLAG
0000      DSKTY8     EQU      0          ;8 INCH DISK TYPE FLAG

;
;
;   ORG      CBIOS
;
0000'     C3 003C'      JP      BOOT          ;STANDARD JUMP TABLE TO
0003'     C3 0056'      BVECTR: JP      WBOOT      ;THE SUBROUTINES OF CBIOS
0006'     C3 0123'      SVECTR: JP      CONST
0009'     C3 0126'      IVECTR: JP      CONIN
000C'     C3 0129'      OVECTR: JP      CONOUT
000F'     C3 012D'      JP      LSTOUT          ;LIST DEVICE VECTOR
0012'     C3 0129'      JP      CONOUT          ;PUNCH DEVICE VECTOR
0015'     C3 0126'      JP      CONIN          ;READER DEVICE VECTOR
0018'     C3 021A'      JP      HOME
001B'     C3 01D8'      JP      SELECT
001E'     C3 0227'      JP      SEEK
0021'     C3 01C8'      JP      SETSEC
0024'     C3 01D3'      JP      SETPTR
0027'     C3 023C'      JP      READ

002A'     C3 0250'      JP      WRITE
002D'     C3 0123'      JP      CONST          ;LIST DEVICE STATUS VECTOR
0030'     C3 01CD'      JP      TRANS

;
;   JUMP VECTORS TO DIRECT PRINTER DRIVERS
;
0033'     C3 014E'      JP      POBUSY          ;LIST DEVICE STATUS
0036'     C3 0158'      JP      POSEND          ;LIST DEVICE OUTPUT

```

5.25" CBIOS VERSION 2.0

```

0039'  C3 015B'      JP      POINP      ;LIST DEVICE INPUT
;
;
;
003C'  AF      BOOT:   XOR      A
003D'  32 0003      LD      (0003H), A      ;RESET IOBYTE TO ZEROS
0040'  32 0338'      LD      (WUNIT), A      ;ZERO SAVE AREA FOR LOGGED DR
;
;      MOVE XEROX ID TO THE SIGN ON MESSAGE
;
0043'  21 00F7      LD      HL,00F7H      ;ADRS OF XEROX ID AFTER BOOT
0046'  11 032A'      LD      DE,XEROXID      ;ADRS OF XEROX ID IN BIOS
0049'  01 0009      LD      BC,09D      ;NUM OF BYTES TO MOVE IN DEC
004C'  ED B0      LDIR      ;MOVE THEM
;
004E'  21 02DF'      LD      HL,SIGNON
0051'  CD 02A2'      CALL     PMSG      ;PRINT SIGNON MESSAGE
0054'  18 59      JR      GOCPM
;
;
;      WBOOT:
0056'  31 035C'      IF      DSKTY5      ;5.25 INCH DISK
0059'  3A 0336'      LD      SP,STACK
005C'  32 0338'      LD      A,(UNIT)      ;SAVE LOGGED DRIVE FOR
005F'  0E 00      LD      (WUNIT),A      ;* LATER USE
0061'  CD 01D8'      LD      C,0
0064'  CD 021A'      CALL     SELECT      ;SELECT UNIT 0
0067'  C2 0106'      CALL     HOME      ;SEEK TRACK ZERO
006A'  21 D480      JP      NZ,BOMB
006D'  01 0803      LD      HL,3480H+BASE
0070'  CD 00F0'      LD      BC,0803H
0073'  21 D400      CALL     RDLOOP      ;READ EVEN SECTORS ON TRK 0
0076'  01 0902      LD      HL,3400H+BASE
0079'  CD 00F0'      LD      BC,0902H
007C'  0E 01      CALL     RDLOOP      ;READ ODD SECTORS ON TRK 0
007E'  CD 0227'      LD      C,1
0081'  C2 0106'      CALL     SEEK      ;SEEK TO TRACK 1
0084'  21 DC80      JP      NZ,BOMB
0087'  01 0901      LD      HL,3C80H+BASE
008A'  CD 00F0'      LD      BC,0901H
008D'  21 DD00      CALL     RDLOOP      ;READ ODD SECTORS ON TRK 1
0090'  01 0902      LD      HL,3D00H+BASE
0093'  CD 00F0'      LD      BC,0902H
0096'  0E 02      CALL     RDLOOP      ;READ EVEN SECTORS ON TRK 1
0098'  CD 0227'      LD      C,2
009B'  20 69      CALL     SEEK      ;SEEK TRACK #2
009D'  21 E580      JR      NZ,BOMB
00A0'  01 0501      LD      HL,4580H+BASE
00A3'  CD 00F0'      LD      BC,0501H
00A6'  21 E600      CALL     RDLOOP      ;READ ODD SECTORS ON TRK 2
00A9'  01 0402      LD      HL,4600H+BASE
00AC'  CD 00F0'      LD      BC,0402H
00AF'  3E C3      CALL     RDLOOP      ;READ EVEN SECTORS ON TRK 2
00B1'  32 0000      GOCPM:  LD      A,0C3H      ;STORE JUMP VCTRS IN RAM
00B4'  21 EA03      LD      (00H),A
00B7'  22 0001      LD      HL,CBIOS+3      ;JP TO CBIOS WARM BOOT AT 00H
00BA'  32 0005      LD      (01H),HL
00BD'  21 DC06      LD      (05H),A
00C0'  22 0006      LD      HL,BDOS      ;JUMP TO BDOS GOES AT 05H
00C3'  32 0038      LD      (06H),HL
00C6'  21 F000      LD      (38H),A
00C9'  22 0039      LD      HL,MONITR      ;JUMP TO MONTR GOES AT 38H
00CC'  01 0080      LD      (39H),HL
00CF'  CD 01D3'      LD      BC,0080H
00D0'  CALL     SETPTR      ;MAKE DISK BUFFER=0080H
00D3'  ENDF      ;END OF 5.25 INCH SECTION
00D6'  IF      DSKTY8      ;8 INCH DISK

```

```

ENDIF
;
;
;      INITIALIZE THE PRINTER
;
00D2'  3E 07      LD      A,07      ;LOAD BAUD RATE
00D4'  D3 0C      OUT     (0CH),A    ;SEND TO SIO CH. B
;
;      SEND A 'RESET' SEQUENCE TO THE PRINTER
;
00D6'  06 03      LD      B,03D     ;NUMBER OF BYTES IN SEQUENCE
00D8'  21 00ED'    LD      HL,INPR2  ;ADRS OF 'RESET' TABLE
00DB'  CD 014E'    INPR1: CALL     POBUSY   ;IS PRINTER READY?
00DE'  38 FB      JR      C,INPR1   ;* REPEAT TILL READY
00E0'  7E         LD      A,(HL)    ;GET THE BYTE
00E1'  D3 05      OUT     (05H),A    ;SEND IT
00E3'  23         INC     HL        ;POINT TO NEXT BYTE
00E4'  10 F5      DJNZ    INPR1     ;REPEAT TILL DONE
;
;
00E6'  3A 0338'    LD      A,(WUNIT) ;SELECT SAVED DRIVE
00E9'  4F         LD      C,A       ;*
00EA'  C3 D400     JP      CCP       ;* JP TO COMMAND CONSOLE PROC
;
;      'RESET' SEQUENCE TABLE FOR PRINTER
;
00ED'  1B         INPR2: DEFB     1BH    ; 'ESC'
00EE'  0D         DEFB     0DH    ; 'CR'
00EF'  50         DEFB     50H    ; 'P'
;
;
00F0'  22 033B'    RDLOOP: LD      (POINTR),HL ;STORE ADDR. PASSED IN HL
00F3'  79         LD      A,C       ;STORE SECT# PASSED IN C
00F4'  32 033A'    LD      (SECTOR),A
00F7'  E5         PUSH    HL
00F8'  C5         PUSH    BC
00F9'  CD 023C'    CALL    READ      ;READ THE SPECIFIED SECTOR
00FC'  C1         POP     BC
00FD'  E1         POP     HL
00FE'  20 06      JR      NZ,BOMB    ;BUMP LOAD ADDRESS BY 256
0100'  24         INC     H
0101'  0C         INC     C
0102'  0C         INC     C         ;BUMP SECTOR# BY 2
0103'  10 EB      DJNZ    RDLOOP
0105'  C9         RET
;
;
0106'  21 010F'    BOMB:  LD      HL,DEAD
0109'  CD 02A2'    CALL    PMSG
010C'  C3 010C'    LOOP:  JP      LOOP
;
010F'  0D 0A      DEAD:  DEFB     CR,LF
0111'  63 61 6E 6E DEFB     'cannot boot CP/M $'
0115'  6F 74 20 62
0119'  6F 6F 74 20
011D'  43 50 2F 4D
0121'  20 24
;
;
;
0123'  C3 F006     CONST:  JP      MONITR+6    ;MONITOR CONSOLE STATUS RTN.
;
0126'  C3 F009     CONIN:  JP      MONITR+9    ;MONITOR CONSOLE INPUT RTN.
;
0129'  79         CONOUT: LD      A,C
012A'  C3 F00C     JP      MONITR+12   ;MONITOR CONSOLE OUTPUT RTN.
;
;

```

```

;*****
;*
;*          LIST OUTPUT DEVICE DRIVER
;*
;*****
;
012D'  CD 015B'  LSTOUT:  CALL    POINP      ;CHECK IF PRINTER HAS DATA
0130'  30 FB      JR      NC,LSTOUT ;REPEAT TILL CLEAR
0132'  CD 014E'  CPBSY:  CALL    POBUSY   ;CHECK IF PRINTER BUSY
0135'  38 FB      JR      C,CPBSY    ;REPEAT TILL READY
0137'  79        LD      A,C        ;GET CHAR FROM C
0138'  CD 0158'  CALL    POSEND    ;PRINT THE CHARACTER
013B'  FE 0A      CP      0AH       ;WAS IT A LINE FEED?
013D'  C0        RET      NZ        ;RETURN IF NOT
013E'  CD 014E'  CPBSY2: CALL    POBUSY   ;IF SO GET PRINTER READY
0141'  38 FB      JR      C,CPBSY2  ;*
0143'  3E 03      LD      A,03      ;LOAD A 'ETX'
0145'  CD 0158'  CALL    POSEND    ;AND PRINT IT
0148'  CD 015B'  WAIT:  CALL    POINP      ;LOOP TILL RECEIVE
014B'  38 FB      JR      C,WAIT    ;* AN 'ACT'
014D'  C9        RET              ;THEN RETURN
;
;
;          PRINTER BUSY ROUTINE
;
014E'  DB 07      POBUSY:  IN      A,(07)   ;READ SIO PORT CH. B
0150'  E6 04      AND      04          ;MASK OUT BITS OF INTEREST
;
0152'  EE 04      XOR      04          ;* TO CHECK PRINTER STATUS
0154'  37        SCF              ;SET CARRY
0155'  C0        RET      NZ        ;RET WITH PRINTER NOT READY
0156'  B7        OR      A          ;* ELSE RESET CARRY
0157'  C9        RET              ;* AND RET WITH PRINTER READY
;
;
;          PRINTER OUTPUT ROUTINE
;
0158'  D3 05      POSEND:  OUT     (05),A    ;SEND THE BYTE
015A'  C9        RET              ;* AND RETURN
;
;
;          PRINTER INPUT STATUS ROUTINE
;
015B'  DB 07      POINP:   IN      A,(07)   ;READ SIO PORT CH. B
015D'  E6 01      AND      01          ;CHECK FOR RECEIVE
015F'  EE 01      XOR      01          ;* CHARACTER AVAILABLE
0161'  37        SCF              ;SET CARRY
0162'  C0        RET      NZ        ;RET WITH NO CHARA AVAIL.
0163'  DB 05      IN      A,(05)      ;* ELSE GET CHARACTER
0165'  B7        OR      A          ;* RESET CARRY
0166'  C9        RET              ;* AND RETURN
;
;
;*****
;*
;*          DISK I/O SUBROUTINES FOR CP/M CBIOS
;*
;*****
;
;          IF          DSKTY5          ;5.25 INCH DISK
;
;
;          SECTOR TRANSLATE TABLE FOR STANDARD
;          1 IN 5 INTERLEAVE FACTOR
;

```

5.25" CBIOS VERSION 2.0


```

01DB' 79          LD      A,C
;
;
01DC' FE 04      IF      DSKTY5      ;5.25" SYSTEM
CP              4              ;IS ALLOWED TO HAVE 4 DRIVES
ENDIF
;
;
IF      DSKTY8      ;8" SYSTEM
ENDIF
;
;
01DE' D0          RET      NC          ;RETURN WITH HL=0 IF C 3
01DF' 32 0336'    LD      (UNIT),A    ;STORE C AS NEW DRIVE UNIT#
01E2' 6F          LD      L,A          ;
01E3' 29          ADD     HL,HL
01E4' 29          ADD     HL,HL
01E5' 29          ADD     HL,HL
01E6' 29          ADD     HL,HL      ;MULTIPLY UNIT# BY 16
01E7' 11 0188'    LD      DE,DPHTAB
01EA' 19          ADD     HL,DE      ;ADD START ADDRESS OF DHP BLK
01EB' C9          RET
01EC' C5          SELEX:  PUSH     BC      ;DO NOT ACTUALLY SEL THE DR
01ED' E5          PUSH     HL          ;SAVE REGISTERS VALUES
01EE' 3A 0336'    LD      A,(UNIT)
01F1' 4F          LD      C,A          ;LOAD C WITH DISK DRIVE NUM
01F2' 3A 0337'    LD      A,(PUNIT)    ;LOAD PREVIOUSLY SELECTED DR
01F5' B9          CP      C          ;COMP WITH CURRENTLY SEL DR
01F6' 28 12      JR      Z,SELEX1      ;DO NOT SELECT IF SAME DRIVE
01F8' 06 00      SELEX2: LD      B,0    ;LD B WTH SEK SPD FOR THIS DR
01FA' CD F01B     CALL     MONITR+27    ;CALL SELCT ROUTNE IN MNITR
01FD' 28 0B      JR      Z,SELEX1
01FF' CD 027C'    CALL     REPORT      ;CALL ERROR ROUTINE
0202' 20 0F      JR      NZ,SELEX3
0204' 3A 0336'    LD      A,(UNIT)    ;SAVE AS NEXT DRIVE
0207' 4F          LD      C,A          ;LOAD DR TO BE SELECTED IN C
0208' 18 EE      JR      SELEX2      ;
020A' 3A 0336'    SELEX1: LD      A,(UNIT) ;LOAD DRIVE JUST SELECTED
020D' 32 0337'    LD      (PUNIT),A    ;STOR IT AS A PREVIOUS DRIVE
0210' E1          POP     HL
0211' C1          POP     BC          ;RESTORE REGISTERS
0212' C9          RET                ;EXT IF SELECTED SUCCESSFULLY
0213' AF          SELEX3: XOR      A
0214' 32 0336'    LD      (UNIT),A
0217' C3 0000     JP      OH          ;DISAB FURTHER BIOS CALLS BY
;                                     ;INDICATING SEL ERROR TO BDOS
;
;
;
021A' CD 01EC'    HOME:  CALL     SELEX      ;FIND OUT IF DR IS SELECTED
021D' CD F01E     CALL     MONITR+30      ;CALL HOME ROUTINE IN MONITOR
0220' C8          RET      Z          ;RETURN IF ALL WENT WELL
0221' CD 027C'    CALL     REPORT
0224' 28 F4      JR      Z,HOME      ;RE-TRY HOME IF ERR INDICATED
0226' C9          RET
;
0227' CD 01EC'    SEEK:  CALL     SELEX      ;FIND OUT IF DR IS SELECTED
022A' 79          LD      A,C          ;GET TRACK # FROM C
022B' 32 0339'    LD      (TRACK),A
022E' CD F021     CALL     MONITR+33      ;CALL SEEK ROUTINE IN MONITOR
0231' C8          RET      Z          ;EXIT IF NO ERRORS INDICATED
0232' CD 027C'    CALL     REPORT      ;REPORT SEEK ERROR TO CONSOLE
0235' C0          RET      NZ        ;RETURN PERMANENT ERR UNLESS
0236' 3A 0339'    LD      A,(TRACK)    ; RE-TRY REQUEST IS INDICATED
0239' 4F          LD      C,A
023A' 18 EB      JR      SEEK
;

```

```

;
;
023C'  CD 01EC'  READ:  CALL  SELEX      ;FIND OUT IF DR IS SELECTED
023F'  2A 033B'  LD      HL,(POINTR)
0242'  3A 033A'  LD      A,(SECTOR)
0245'  4F        LD      C,A
0246'  CD F024   CALL     MONITR+36  ;CALL READ ROUTINE IN MONITOR
0249'  C8        RET      Z      ;RETURN IF NO ERRORS
024A'  CD 027C'  CALL     REPORT  ;REPORT DISK ERROR TO CONSOLE
024D'  28 ED     JR      Z,READ  ;RE-TRY READ IF INDICATED
024F'  C9        RET

;
;
0250'  CD 01EC'  WRITE:  CALL  SELEX      ;FIND OUT IF DR IS SELECTED
0253'  2A 033B'  LD      HL,(POINTR)
0256'  3A 033A'  LD      A,(SECTOR)
0259'  4F        LD      C,A
025A'  CD F027   CALL     MONITR+39  ;CALL WRIT ROUTINE IN MONITOR
025D'  C8        RET      Z      ;RETURN IF NO ERRORS
025E'  5F        LD      E,A      ;SAVE 1771 I/O STATUS FLAG
025F'  17        RLA
0260'  17        RLA
0261'  30 12     JR      NC,WRIT1  ;CRY CONTAINS WRIT PROT STAT
0263'  21 02AD'  LD      HL,DSKMSG ;CONT IF NOT WRITE PROTECTED
0266'  CD 02A2'  CALL     PMSG      ;PRINT 'disk'
0269'  21 02CF'  LD      HL,WRterr
026C'  CD 02A2'  CALL     PMSG      ;PRINT 'write protected'
026F'  CD 0299'  CALL     REP3      ;WAIT FOR CONSOLE INPUT
0272'  28 DC     JR      Z,WRITE  ;RETRY IF INDICATED
0274'  C9        RET
0275'  7B        WRIT1:  LD      A,E  ;RETRIEVE SAVED 1771 STATUS
0276'  CD 027C'  CALL     REPORT  ;REPORT DISK ERROR TO CONSOLE
0279'  28 D5     JR      Z,WRITE  ;RE-TRY WRITE IF INDICATED
027B'  C9        RET             ;ELSE RETURN PERMANENT ERROR

;
;
; ON ENTRY (A) = 1771 I/O STATUS FLAG
;
;
027C'  F5        REPORT:  PUSH     AF      ;SAVE 1771 I/O STATUS
027D'  21 02AD'  LD      HL,DSKMSG
0280'  CD 02A2'  CALL     PMSG      ;PRINT 'disk '
0283'  F1        POP      AF      ;RETRIEVE SAVED STATUS
0284'  17        RLA
0285'  38 0C     JR      C,REP2    ;TST FST FOR DR-NOT-READY ERR
0287'  21 02B5'  LD      HL,ERRMSG ;JUMP IF THAT IS THE PROBLEM
028A'  CD 02A2'  CALL     PMSG      ;OTHER TYPE OF ERROR SO ---
028D'  18 0A     JR      REP3      ;PRINT 'error '
028F'  3E 01     REP1:  LD      A,1  ;GET CONSOLE INPUT
                                ;SET UP A NZERO COND FOR RET

0291'  B7        OR      A      ;RET PERM ERR INDICATION IN A
0292'  C9        RET

;
;
0293'  21 02BD'  REP2:  LD      HL,RDYMMSG
0296'  CD 02A2'  CALL     PMSG      ;PRINT DISK-NOT-READY MESSAGE
0299'  CD 0009'  REP3:  CALL     IVECTR ;AND WAIT FOR CONSOLE INPUT
029C'  FE 03     CP      'C'-64
029E'  28 EF     JR      Z,REP1
02A0'  AF        XOR      A      ;RET A=0 IF SOMETHING OTR THN
02A1'  C9        RET             ;CONT-C WAS TYPED AT THE CONS

;
;
; CHARACTER STRING OUTPUT ROUTINE. PRINTS ASCII DATA
; POINTED TO BY HL UNTIL A DOLLAR SIGN IS ENCOUNTERED

```

02A2'	7E	PMSG:	LD	A,(HL)	;HL POINTS TO ASCII STRING
02A3'	FE 24		CP	'\$'	
02A5'	23		INC	HL	
02A6'	C8		RET	Z	
02A7'	4F		LD	C,A	;PRNT CHAR IF NOT DOLLAR SIGN
02A8'	CD 000C'		CALL	OVECTR	
02AB'	18 F5		JR	PMSG	
000A		LF	EQU	OAH	;LINE FEED
000D		CR	EQU	ODH	;CARRIAGE RETURN
02AD'	0D 0A	DSKMSG:	DEFB	CR,LF	
02AF'	64 69 73 6B		DEFM	'disk \$'	
02B3'	20 24				
02B5'	65 72 72 6F	ERRMSG:	DEFM	'error \$'	
02B9'	72 20 20 24				
02BD'	64 72 69 76	RDYMSG:	DEFM	'drive not ready -\$'	
02C1'	65 20 6E 6F				
02C5'	74 20 72 65				
02C9'	61 64 79 20				
02CD'	2D 24				
02CF'	77 72 69 74	WRTErr:	DEFM	'write protected\$'	
02D3'	65 20 70 72				
02D7'	6F 74 65 63				
02DB'	74 65 64 24				
02DF'	0D 0A	SIGNON:	DEFB	CR,LF	
02E1'	43 4F 50 59		DEFM	'COPYRIGHT (C) 1981, XEROX CORPORATION'	
02E5'	52 49 47 48				
02E9'	54 20 28 43				
02ED'	29 20 31 39				
02F1'	38 31 2C 20				
02F5'	58 45 52 4F				
02F9'	58 20 43 4F				
02FD'	52 50 4F 52				
0301'	41 54 49 4F				
0305'	4E				
0306'	0D 0A		DEFB	CR,LF	
0308'	0D 0A		DEFB	CR,LF	
030A'	43 50 2F 4D		DEFM	'CP/M REG. TM 2.2 SY 2.0 2-294 '	
030E'	20 52 45 47				
0312'	2E 20 54 4D				
0316'	20 32 2E 32				
031A'	20 20 53 59				
031E'	20 32 2E 30				
0322'	20 20 32 2D				
0326'	32 39 34 20				
032A'	20 20 20 20	XEROXID:	DEFM	' '	
032E'	20 20 20 20				
0332'	20				
0333'	0D 0A	CRLF:	DEFB	CR,LF	
0335'	24		DEFB	'\$'	
0336'		UNIT:	DEFS	1	
0337'		PUNIT:	DEFS	1	
0338'		WUNIT:	DEFS	1	
0339'		TRACK:	DEFS	1	
033A'		SECTOR:	DEFS	1	
033B'		POINTR:	DEFS	1	
033C'			DEFS	32	
035C'		STACK:	DEFS	1	;LOCAL STACK FOR WARM BOOT


```

.Z80
.SFCOND
;*****
;*
;*  -- CUSTOM BIOS FOR CP/M VERSION 2.2  --
;*      8-INCH DISK VERSION
;*
;*      APRIL 1981
;*
;*      CBIOS FOR XEROX CP/M DISK
;*  COMBINED VERSION FOR 5.25" AND 8" - JUNE 1981
;*
;*****
;
;      ASEG
;
003C      MSIZE      EQU      60          ;MEMORY CAPACITY IN KBYTES
F000      MONITR     EQU      0F000H     ;BASE OF SYSTEM MONITOR

0028      EXTRA     EQU      MSIZE-20
A000      BASE      EQU      EXTRA*1024

D400      CCP       EQU      3400H+BASE  ;CONSOLE COMMAND PROCESSOR
DC06      BDOS      EQU      3C06H+BASE  ;OPERATING SYSTEM ENTRY POINT
EA00      CBIOS     EQU      4A00H+BASE  ;BASE OF CUSTOM BIOS

;
;      EQUATES TO SELECT THE CONDITIONAL ASSEMBLY
;      FOR 5.25 OR 8 INCH DISKS
;
;      THE EQUATES DSKTY5 OR DSKTY8 ARE USED FOR
;      CONDITIONAL ASSEMBLY CONTROL
;
;      ONE CONDITIONAL ASSEMBLY FLAG SHOULD BE ON
;      AND THE OTHER SHOULD BE OFF AT ALL TIMES.
;
0000      DSKTY5     EQU      0          ;5.25 INCH DISK TYPE FLAG
0001      DSKTY8     EQU      1          ;8 INCH DISK TYPE FLAG

;
;      ORG      CBIOS
;
0000'     C3 003C'     JP      BOOT      ;STANDARD JUMP TABLE TO
0003'     C3 0056'     BVECTR: JP      WBOOT ;THE SUBROUTINES OF CBIOS
0006'     C3 0107'     SVECTR: JP      CONST
0009'     C3 010A'     IVECTR: JP      CONIN
000C'     C3 010D'     OVECTR: JP      CONOUT
000F'     C3 0111'     JP      LSTOUT     ;LIST DEVICE VECTOR
0012'     C3 010D'     JP      CONOUT     ;PUNCH DEVICE VECTOR
0015'     C3 010A'     JP      CONIN      ;READER DEVICE VECTOR
0018'     C3 01E6'     JP      HOME
001B'     C3 01A4'     JP      SELECT
001E'     C3 01F3'     JP      SEEK
0021'     C3 0194'     JP      SETSEC
0024'     C3 019F'     JP      SETPTR
0027'     C3 0208'     JP      READ

002A'     C3 021C'     JP      WRITE
002D'     C3 0107'     JP      CONST      ;LIST DEVICE STATUS VECTOR
0030'     C3 0199'     JP      TRANS

;
;      JUMP VECTORS TO DIRECT PRINTER DRIVERS
;
0033'     C3 0132'     JP      POBUSY     ;LIST DEVICE STATUS

```

8.00" CBIOS VERSION 2.0

```

0036'  C3 013C'      JP      POSEND      ;LIST DEVICE OUTPUT
0039'  C3 013F'      JP      POINP       ;LIST DEVICE INPUT
;
;
;
003C'  AF            BOOT:  XOR      A              ;RESET IOBYTE TO ZEROS
003D'  32 0003      LD      (0003H),A
0040'  32 0304'      LD      (WUNIT),A      ;ZERO SAVE AREA FOR LOGGED DR
;
;
;      MOVE XEROX ID TO THE SIGN ON MESSAGE
0043'  21 00F7      LD      HL,00F7H      ;ADRS OF XEROX ID AFTER BOOT
0046'  11 02F6'      LD      DE,XEROXID    ;ADRS OF XEROX ID IN BIOS
0049'  01 0009      LD      BC,09D        ;NUM OF BYTES TO MOVE IN DEC
004C'  ED B0        LDIR                     ;MOVE THEM
;
004E'  21 02AB'      LD      HL,SIGNON
0051'  CD 026E'      CALL     PMSG          ;PRINT SIGNON MESSAGE
0054'  18 3D        JR      GOCPM
;
;
;      IF      DSKTY5      ;5.25 INCH DISK
;      ENDIF      ;END OF 5.25 INCH SECTION
;      IF      DSKTY8      ;8 INCH DISK
0056'  31 0328'      WBOOT: LD      SP,STACK
0059'  3A 0302'      LD      A,(UNIT)      ;SAVE LOGGED DRIVE FOR
005C'  32 0304'      LD      (WUNIT),A      ;* LATER USE
005F'  0E 00        LD      C,0
0061'  CD 01A4'      CALL     SELECT        ;SELECT UNIT 0
0064'  CD 01E6'      CALL     HOME         ;SEEK TRACK ZERO
0067'  21 D400      LD      HL,3400H+BASE
006A'  01 0D02      LD      BC,0D02H
006D'  CD 00D4'      CALL     RDLOOP        ;READ EVEN SECTORS ON TRK 0
0070'  21 D480      LD      HL,3480H+BASE
0073'  01 0C03      LD      BC,0C03H
0076'  CD 00D4'      CALL     RDLOOP        ;READ ODD SECTORS ON TRK 0
0079'  0E 01        LD      C,1
007B'  CD 01F3'      CALL     SEEK          ;SEEK TO TRACK 1
007E'  C2 00EA'      JP      NZ,BOMB
0081'  21 E080      LD      HL,4080H+BASE
0084'  01 0A01      LD      BC,0A01H
0087'  CD 00D4'      CALL     RDLOOP        ;READ ODD SECTORS ON TRK 1
008A'  21 E100      LD      HL,4100H+BASE
008D'  01 0902      LD      BC,0902H
0090'  CD 00D4'      CALL     RDLOOP        ;READ EVEN SECTORS ON TRK 1
0093'  3E C3        GOCPM: LD      A,0C3H      ;STORE JUMP VECTORS IN RAM
0095'  32 0000      LD      (00H),A
0098'  21 EA03      LD      HL,CBIOS+3      ;JP TO CBIOS WARM BOOT AT 00H
009B'  22 0001      LD      (01H),HL
;
009E'  32 0005      LD      (05H),A
00A1'  21 DC06      LD      HL,BDOS        ;JUMP TO BDOS GOES AT 05H
00A4'  22 0006      LD      (06H),HL
00A7'  32 0038      LD      (38H),A
00AA'  21 F000      LD      HL,MONITR      ;JUMP TO MONTR GOES AT 38H
00AD'  22 0039      LD      (39H),HL
00B0'  01 0080      LD      BC,0080H
00B3'  CD 019F'      CALL     SETPTR        ;MAKE DISK BUFFER=0080H
;
;      ENDIF
;
;      INITIALIZE THE PRINTER
;
00B6'  3E 07        LD      A,07          ;LOAD BAUD RATE
00B8'  D3 0C        OUT     (0CH),A        ;SEND TO SIO CH. B
;
;      SEND A 'RESET' SEQUENCE TO THE PRINTER
;

```

```

00BA' 06 03          LD      B,03D          ;NUMBER OF BYTES IN SEQUENCE
00BC' 21 00D1'       LD      HL,INPR2       ;ADRS OF 'RESET' TABLE
00BF' CD 0132'       INPR1: CALL    POBUSY    ;IS PRINTER READY?
00C2' 38 FB          JR      C,INPR1       ;* REPEAT TILL READY
00C4' 7E             LD      A,(HL)        ;GET THE BYTE
00C5' D3 05          OUT     (05H),A       ;SEND IT
00C7' 23             INC     HL           ;POINT TO NEXT BYTE
00C8' 10 F5          DJNZ    INPR1        ;REPEAT TILL DONE
;
;
00CA' 3A 0304'       LD      A,(WUNIT)     ;SELECT SAVED DRIVE
00CD' 4F             LD      C,A           ;*
00CE' C3 D400        JP      CCP          ;* JP TO COMMAND CONSOLE PROC
;
;          'RESET' SEQUENCE TABLE FOR PRINTER
;
00D1' 1B             INPR2: DEFB     1BH     ; 'ESC'
00D2' 0D             DEFB     0DH     ; 'CR'
00D3' 50             DEFB     50H     ; 'P'
;
;
00D4' 22 0307'       RDLOOP: LD      (POINTR),HL ;STORE ADDR. PASSED IN HL
00D7' 79             LD      A,C         ;
00D8' 32 0306'       LD      (SECTOR),A   ;STORE SECT# PASSED IN C
00DB' E5             PUSH    HL          ;
00DC' C5             PUSH    BC          ;
00DD' CD 0208'       CALL    READ        ;READ THE SPECIFIED SECTOR
00E0' C1             POP     BC          ;
00E1' E1             POP     HL          ;
00E2' 20 06          JR      NZ,BOMB     ;
00E4' 24             INC     H           ;BUMP LOAD ADDRESS BY 256
00E5' 0C             INC     C           ;
00E6' 0C             INC     C           ;BUMP SECTOR# BY 2
00E7' 10 EB          DJNZ    RDLOOP      ;
00E9' C9             RET               ;
;
;
00EA' 21 00F3'       BOMB:  LD      HL,DEAD
00ED' CD 026E'       CALL    PMSG
;
00F0' C3 00F0'       LOOP:   JP      LOOP
;
00F3' 0D 0A          DEAD:  DEFB     CR,LF
00F5' 63 61 6E 6E    DEFM     'cannot boot CP/M $'
00F9' 6F 74 20 62
00FD' 6F 6F 74 20
0101' 43 50 2F 4D
0105' 20 24
;
;
;
0107' C3 F006        CONST: JP      MONITR+6 ;MONITOR CONSOLE STATUS RTN.
;
010A' C3 F009        CONIN:  JP      MONITR+9 ;MONITOR CONSOLE INPUT RTN.
;
010D' 79             CONOUT: LD      A,C
010E' C3 F00C        JP      MONITR+12 ;MONITOR CONSOLE OUTPUT RTN.
;
;
;*****
;*
;*          LIST OUTPUT DEVICE DRIVER
;*
;*****
;
0111' CD 013F'       LSTOUT: CALL    POINP    ;CHECK IF PRINTER HAS DATA
0114' 30 FB          JR      NC,LSTOUT    ;REPEAT TILL CLEAR

```



```

0116'   CD 0132'   CPBSY:   CALL   POBUSY   ;CHECK IF PRINTER BUSY
0119'   38 FB      JR       C,CPBSY   ;REPEAT TILL READY
011B'   79         LD       A,C       ;GET CHAR FROM C
011C'   CD 013C'   CALL   POSEND   ;PRINT THE CHARACTER
011F'   FE 0A      CP       OAH      ;WAS IT A LINE FEED?
0121'   C0         RET       NZ       ;RETURN IF NOT
0122'   CD 0132'   CPBSY2:  CALL   POBUSY   ;IF SO GET PRINTER READY
0125'   38 FB      JR       C,CPBSY2 ;*
0127'   3E 03      LD       A,03     ;LOAD A 'ETX'
0129'   CD 013C'   CALL   POSEND   ;AND PRINT IT
012C'   CD 013F'   WAIT:    CALL   POINP   ;LOOP TILL RECEIVE
012F'   38 FB      JR       C,WAIT   ;* AN 'ACT'
0131'   C9         RET              ;THEN RETURN
;
;
; PRINTER BUSY ROUTINE
;
0132'   DB 07      POBUSY:  IN       A,(07) ;READ SIO PORT CH. B
0134'   E6 04      AND      04         ;MASK OUT BITS OF INTEREST
0136'   EE 04      XOR      04         ;* TO CHECK PRINTER STATUS
0138'   37         SCF              ;SET CARRY
0139'   C0         RET       NZ       ;RET WITH PRINTER NOT READY
013A'   B7         OR        A         ;* ELSE RESET CARRY
013B'   C9         RET              ;* AND RET WITH PRINTER READY
;
;
; PRINTER OUTPUT ROUTINE
;
013C'   D3 05      POSEND:  OUT      (05),A ;SEND THE BYTE
013E'   C9         RET              ;* AND RETURN
;
;
; PRINTER INPUT STATUS ROUTINE
;
013F'   DB 07      POINP:   IN       A,(07) ;READ SIO PORT CH. B
0141'   E6 01      AND      01         ;CHECK FOR RECEIVE
0143'   EE 01      XOR      01         ;* CHARACTER AVAILABLE
0145'   37         SCF              ;SET CARRY
0146'   C0         RET       NZ       ;RET WITH NO CHARA AVAIL.
0147'   DB 05      IN       A,(05)     ;* ELSE GET CHARACTER
0149'   B7         OR        A         ;* RESET CARRY
014A'   C9         RET              ;* AND RETURN
;
;
;*****
;* DISK I/O SUBROUTINES FOR CP/M CBIOS *
;* *****
;
;
IF      DSKTY5      ;5.25 INCH DISK
ENDIF   ;END OF 5.25 INCH SECTION
IF      DSKTY8      ;8 INCH DISK
;
;
; SECTOR TRANSLATE TABLE FOR STANDARD
; 1 IN6 INTERLEAVE FACTOR
;
014B'   01 07 0D 13 SECTAB:  DEFB      1,7,13,19
014F'   19 05 0B 11      DEFB      25,5,11,17
0153'   17 03 09 0F      DEFB      23,3,9,15
0157'   15 02 08 0E      DEFB      21,2,8,14
015B'   14 1A 06 0C      DEFB      20,26,6,12
015F'   12 18 04 0A      DEFB      18,24,4,10

```

8.00" CBIOS VERSION 2.0

```

0163' 10 16          DEFB      16,22
;
;
;      DISK PARAMETER BLOCK FOR STANDARD 8" FLOPPY
;
0165' 001A          DPBLK:  DEFW      26          ;SECTORS PER TRACK
0167' 03            DEFB      3          ;BLOCK SHIFT CONST.
0168' 07            DEFB      7          ;BLOCK MASK CONST.
0169' 00            DEFB      0          ;EXTENT MASK CONST.
016A' 00F2          DEFW      242         ;MAX BLOCK#
016C' 003F          DEFW      63          ;MAX DIRECTORY ENTRY#
016E' C0            DEFB      11000000B   ;ALLOCATION MASK MSB
016F' 00            DEFB      00000000B   ;'          ' LSB
0170' 0010          DEFW      16          ;CHECK SIZE
0172' 0002          DEFW      2          ;RESERVED TRACKS
;
;
;      DISK PARAMETER HEADERS FOR A 2 DISK SYSTEM
;      THE LAST TWO REMOVED FOR SPACE.
;
0174' 014B' 0000    DPHTAB:  DEFW      SECTAB,0000H ;DPH FOR UNIT 0
0178' 0000 0000    DEFW      0000H,0000H
017C' 0329' 0165'   DEFW      DIRBUF,DPBLK
0180' 03C9' 03A9'   DEFW      CHK0,ALLO
;
0184' 014B' 0000    DEFW      SECTAB,0000H ;DPH FOR UNIT 1
0188' 0000 0000    DEFW      0000H,0000H
018C' 0329' 0165'   DEFW      DIRBUF,DPBLK
0190' 03F9' 03D9'   DEFW      CHK1,ALL1
;
;
;      ENDIF          ;END OF 8 INCH SECTION
;
;
;
0194' 79            SETSEC:  LD        A,C
0195' 32 0306'      LD        (SECTOR),A ;STORE SECTOR NUMBER PASSED
0198' C9            RET      ; VIA BC
;
;
;
0199' EB            TRANS:  EX        DE,HL ;ADD TRANSLATION TABLE ADDR
019A' 09            ADD      HL,BC ; PASSED IN DE TO SEC # IN BC
019B' 6E            LD        L,(HL)
019C' 26 00          LD        H,0 ;LOOKUP PHYSICAL SEC NUMBER
019E' C9            RET      ; AND RETURN IT IN HL
;
;
;
019F' ED 43 0307'   SETPTR:  LD        (POINTR),BC ;STORE DATA POINTER PASSED
01A3' C9            RET      ; VIA BC
;
;
;
01A4' 21 0000       SELECT:  LD        HL,0 ;PREP TO CHECK FOR MAX UNIT#
01A7' 79            LD        A,C
;
;
;
;      IF            DSKTY5 ;5.25" SYSTEM
;      ENDIF
;
;
;
;      IF            DSKTY8 ;8" SYSTEM
01A8' FE 02         CP        2 ;IS ALLOWED TO HAVE 2 DRIVES
;      ENDIF
;
;
;
01AA' D0            RET      NC ;RETURN WITH HL=0 IF C 3
01AB' 32 0302'      LD        (UNIT),A ;STORE C AS NEW DRIVE UNIT#
01AE' 6F            LD        L,A ;

```

01AF'	29		ADD	HL,HL	
01B0'	29		ADD	HL,HL	
01B1'	29		ADD	HL,HL	
01B2'	29		ADD	HL,HL	;MULTIPLY UNIT# BY 16
01B3'	11 0174'		LD	DE,DPHTAB	
01B6'	19		ADD	HL,DE	;ADD START ADDRESS OF DHP BL
01B7'	C9		RET		;DO NOT ACTUALLY SEL THE DR
01B8'	C5	SELEX:	PUSH	BC	;SAVE REGISTERS VALUES
01B9'	E5		PUSH	HL	
01BA'	3A 0302'		LD	A,(UNIT)	
01BD'	4F		LD	C,A	;LOAD C WITH DISK DR NUMBER
01BE'	3A 0303'		LD	A,(PUNIT)	;LOAD PREVIOUSLY SELECTED DR
01C1'	B9		CP	C	;COMPARE WTH CURRENTLY SEL DR
01C2'	28 12		JR	Z,SELEX1	;DO NOT SELECT IF SAME DRIVE
01C4'	06 00	SELEX2:	LD	B,0	;LD B WTH SEK SPD FOR THIS DR
01C6'	CD F01B		CALL	MONITR+27	;CALL SEL ROUTINE IN MONITOR
01C9'	28 0B		JR	Z,SELEX1	
01CB'	CD 0248'		CALL	REPORT	;CALL ERROR ROUTINE
01CE'	20 0F		JR	NZ,SELEX3	
01D0'	3A 0302'		LD	A,(UNIT)	;SAVE AS NEXT DRIVE
01D3'	4F		LD	C,A	;LOAD DR TO BE SELECTED IN C
01D4'	18 EE		JR	SELEX2	
01D6'	3A 0302'	SELEX1:	LD	A,(UNIT)	;LOAD DRIVE JUST SELECTED
01D9'	32 0303'		LD	(PUNIT),A	;STOR IT AS A PREVIOUS DRIVE
01DC'	E1		POP	HL	
01DD'	C1		POP	BC	;RESTORE REGISTERS
01DE'	C9		RET		;EXT IF SELECTED SUCCESSFULLY
01DF'	AF	SELEX3:	XOR	A	
01E0'	32 0302'		LD	(UNIT),A	
01E3'	C3 0000		JP	OH	;DISAB FURTHER BIOS CALLS BY
			RET		; INDICATING SEL ERR TO BDOS
01E6'	CD 01B8'	HOME:	CALL	SELEX	;FIND OUT IF DR IS SELECTED
01E9'	CD F01E		CALL	MONITR+30	;CALL HOME ROUTINE IN MONITOR
01EC'	C8		RET	Z	;RETURN IF ALL WENT WELL
01ED'	CD 0248'		CALL	REPORT	
01F0'	28 F4		JR	Z,HOME	;RE-TRY HOME IF ERR INDICATED
01F2'	C9		RET		
01F3'	CD 01B8'	SEEK:	CALL	SELEX	;FIND OUT IF DR IS SELECTED
01F6'	79		LD	A,C	;GET TRACK # FROM C
01F7'	32 0305'		LD	(TRACK),A	
01FA'	CD F021		CALL	MONITR+33	;CALL SEEK ROUTINE IN MONITOR
01FD'	C8		RET	Z	;EXIT IF NO ERRORS INDICATED
01FE'	CD 0248'		CALL	REPORT	;REPORT SEEK ERROR TO CONSOLE
0201'	C0		RET	NZ	;RETURN PERMANENT ERR UNLESS
0202'	3A 0305'		LD	A,(TRACK)	; RE-TRY REQUEST IS INDICATED
0205'	4F		LD	C,A	
0206'	18 EB		JR	SEEK	
0208'	CD 01B8'	READ:	CALL	SELEX	;FIND OUT IF DR IS SELECTED
020B'	2A 0307'		LD	HL,(POINTR)	
020E'	3A 0306'		LD	A,(SECTOR)	
0211'	4F		LD	C,A	
0212'	CD F024		CALL	MONITR+36	;CALL READ ROUTINE IN MONITOR
0215'	C8		RET	Z	;RETURN IF NO ERRORS
0216'	CD 0248'		CALL	REPORT	;REPORT DISK ERROR TO CONSOLE
0219'	28 ED		JR	Z,READ	;RE-TRY READ IF INDICATED
021B'	C9		RET		

```

021C'  CD 01B8'  WRITE:  CALL  SELEX      ;FIND OUT IF DR IS SELECTED
021F'  2A 0307'  LD      HL,(POINTR)
0222'  3A 0306'  LD      A,(SECTOR)
0225'  4F        LD      C,A
0226'  CD F027   CALL    MONITR+39  ;CALL WRIT ROUTINE IN MONITOR
0229'  C8        RET      Z      ;RETURN IF NO ERRORS
022A'  5F        LD      E,A    ;SAVE 1771 I/O STATUS FLAG
022B'  17        RLA
022C'  17        RLA      ;CRY CONTAINS WRITE PROT STAT
022D'  30 12     JR      NC,WRIT1 ;CONT IF NOT WRITE PROTECTED
022F'  21 0279'  LD      HL,DSKMSG
0232'  CD 026E'  CALL    PMSG      ;PRINT 'disk'
0235'  21 029B'  LD      HL,WRERR
0238'  CD 026E'  CALL    PMSG      ;PRINT 'write protected'
023B'  CD 0265'  CALL    REP3      ;WAIT FOR CONSOLE INPUT
023E'  28 DC     JR      Z,WRITE   ;RETRY IF INDICATED
0240'  C9        RET
0241'  7B        WRIT1: LD      A,E      ;RETRIEVE SAVED 1771 STATUS
0242'  CD 0248'  CALL    REPORT    ;REPORT DISK ERROR TO CONSOLE
0245'  28 D5     JR      Z,WRITE   ;RE-TRY WRITE IF INDICATED
0247'  C9        RET      ;ELSE RETURN PERMANENT ERROR

;
;
;      ON ENTRY (A) = 1771 I/O STATUS FLAG
;
;
0248'  F5        REPORT: PUSH    AF      ;SAVE 1771 I/O STATUS
0249'  21 0279'  LD      HL,DSKMSG
024C'  CD 026E'  CALL    PMSG      ;PRINT 'disk '
024F'  F1        POP      AF      ;RETRIEVE SAVED STATUS
0250'  17        RLA      ;TST FST FOR DR-NOT-READY ERR
0251'  38 0C     JR      C,REP2    ;JUMP IF THAT IS THE PROBLEM
0253'  21 0281'  LD      HL,ERRMSG ;OTHER TYPE OF ERROR SO ---
0256'  CD 026E'  CALL    PMSG      ;PRINT 'error '
0259'  18 0A     JR      REP3      ;GET CONSOLE INPUT
025B'  3E 01     REP1:  LD      A,l    ;SET UP A NZERO COND FOR RET
025D'  B7        OR      A      ;RET PERM ERR INDICATION IN A
025E'  C9        RET

;
;
025F'  21 0289'  REP2:  LD      HL,RDYMMSG
0262'  CD 026E'  CALL    PMSG      ;PRINT DISK-NOT-READY MESSAGE
0265'  CD 0009'  REP3:  CALL    IVECTR ;AND WAIT FOR CONSOLE INPUT
0268'  FE 03     CP      'C'-64
026A'  28 EF     JR      Z,REP1
026C'  AF        XOR      A      ;RET A=0 IF SOMETHING OTR THN
026D'  C9        RET      ;CONT-C WAS TYPED AT THE CONS

;
;
;      CHARACTER STRING OUTPUT ROUTINE. PRINTS ASCII DATA
;      POINTED TO BY HL UNTIL A DOLLAR SIGN IS ENCOUNTERED
;
026E'  7E        PMSG:  LD      A,(HL) ;HL POINTS TO ASCII STRING
026F'  FE 24     CP      '$'
0271'  23        INC      HL
0272'  C8        RET      Z
0273'  4F        LD      C,A      ;PRNT CHAR IF NOT DOLLAR SIGN
0274'  CD 000C'  CALL    OVECTR
0277'  18 F5     JR      PMSG

;
;
;
000A    LF      EQU      OAH      ;LINE FEED
000D    CR      EQU      ODH      ;CARRIAGE RETURN

```

```

0279' 0D 0A          DSKMSG:  DEFB      CR,LF
027B' 64 69 73 6B    DEFM      'disk $'
027F' 20 24
0281' 65 72 72 6F    ERRMSG:  DEFM      'error  $'
0285' 72 20 20 24
0289' 64 72 69 76    RDYMSG:  DEFM      'drive not ready -$'
028D' 65 20 6E 6F
0291' 74 20 72 65
0295' 61 64 79 20
0299' 2D 24
029B' 77 72 69 74    WRTERR:  DEFM      'write protected$'
029F' 65 20 70 72
02A3' 6F 74 65 63
02A7' 74 65 64 24
02AB' 0D 0A          SIGNON:  DEFB      CR,LF
02AD' 43 4F 50 59    DEFM      'COPYRIGHT (C) 1981, XEROX CORPORATION'
02B1' 52 49 47 48
02B5' 54 20 28 43
02B9' 29 20 31 39
02BD' 38 31 2C 20
02C1' 58 45 52 4F
02C5' 58 20 43 4F
02C9' 52 50 4F 52
02CD' 41 54 49 4F
02D1' 4E
02D2' 0D 0A          DEFB      CR,LF
02D4' 0D 0A          DEFB      CR,LF
02D6' 43 50 2F 4D    DEFM      'CP/M REG. TM 2.2  SY 2.0  2-294 '
02DA' 20 52 45 47
02DE' 2E 20 54 4D
02E2' 20 32 2E 32
02E6' 20 20 53 59
02EA' 20 32 2E 30
02EE' 20 20 32 2D
02F2' 32 39 34 20
02F6' 20 20 20 20    XEROXID:  DEFM      '
02FA' 20 20 20 20
02FE' 20
02FF' 0D 0A          CRLF:    DEFB      CR,LF
0301' 24             DEFB      '$'
;
;
0302' UNIT:         DEFS      1
0303' PUNIT:        DEFS      1
0304' WUNIT:        DEFS      1
0305' TRACK:        DEFS      1
0306' SECTOR:       DEFS      1
0307' POINTR:       DEFS      1
0308'              DEFS      32
0328' STACK:       DEFS      1          ;LOCAL STACK FOR WARM BOOT
;
;*****
;*
;*          DISK I/O BUFFERS FOR BDOS FILE HANDLER          *
;*
;*****
;
;
;
0329' DIRBUF:      DEFS      128          ;SCRATCH DIRECTORY BUFFER
;
03A9' ALLO:         DEFS      32          ;UNIT 0 ALLOCATION BUFFER
03C9' CHK0:         DEFS      16          ;UNIT 0 CHECK VECTOR
03D9' ALL1:         DEFS      32          ;UNIT 1 ALLOCATION VECTOR
03F9' CHK1:         DEFS      16          ;UNIT 1 CHECK VECTOR
IF      DSKTY5          ;ONLY FOR 5.25 INCH DISK

```

ENDIF

;
;
;
;
;

END

Macro\$:

Symbols:

ALLO	03A9'	ALL1	03D9'	BASE	A000	BDOS	DC06
BOMB	00EA'	BOOT	003C'	BVECTR	0003'	CBIOS	EA00
CCP	D400	CHK0	03C9'	CHK1	03F9'	CONIN	010A'
CONOUT	010D'	CONST	0107'	CPBSY	0116'	CPBSY2	0122'
CR	000D	CRLF	02FF'	DEAD	00F3'	DIRBUF	0329'
DPBLK	0165'	DPHTAB	0174'	DSKMSG	0279'	DSKTY5	0000
DSKTY8	0001	ERRMSG	0281'	EXTRA	0028	GOCPM	0093'
HOME	01E6'	INPR1	00BF'	INPR2	00D1'	IVECTR	0009'
LF	000A	LOOP	00F0'	LSTOUT	0111'	MONITR	F000
MSIZE	003C	OVECTR	000C'	PMSG	026E'	POBUSY	0132'
POINP	013F'	POINTR	0307'	POSEND	013C'	PUNIT	0303'
RDLOOP	00D4'	RDYMSG	0289'	READ	0208'	REP1	025B'
REP2	025F'	REP3	0265'	REPORT	0248'	SECTAB	014B'
SECTOR	0306'	SEEK	01F3'	SELECT	01A4'	SELEX	01B8'
SELEX1	01D6'	SELEX2	01C4'	SELEX3	01DF'	SETPTR	019F'
SETSEC	0194'	SIGNON	02AB'	STACK	0328'	SVECTR	0006'
TRACK	0305'	TRANS	0199'	UNIT	0302'	WAIT	012C'
WBOOT	0056'	WRIT1	0241'	WRITE	021C'	WRTERR	029B'
WUNIT	0304'	XEROXI	02F6'				

No Fatal error(s)

ZILOG DATA

Zilog and Z80 are trademarks of Zilog, Inc., with whom the publisher is not associated.

Reproduced by permission © 1979, 1980, 1981 Zilog, Inc. This material shall not be reproduced without the written consent of Zilog, Inc.

Z8400 Z80[®] CPU Central Processing Unit



Product Specification

March 1981

Features

- The instruction set contains 158 instructions. The 78 instructions of the 8080A are included as a subset; 8080A software compatibility is maintained.
- Six MHz, 4 MHz and 2.5 MHz clocks for the Z80B, Z80A, and Z80 CPU result in rapid instruction execution with consequent high data throughput.
- The extensive instruction set includes string, bit, byte, and word operations. Block searches and block transfers together with indexed and relative addressing result in the most powerful data handling capabilities in the microcomputer industry.
- The Z80 microprocessors and associated family of peripheral controllers are linked by a vectored interrupt system. This system may be daisy-chained to allow implementation of a priority interrupt scheme. Little, if any, additional logic is required for daisy-chaining.
- Duplicate sets of both general-purpose and flag registers are provided, easing the design and operation of system software through single-context switching, background-foreground programming, and single-level interrupt processing. In addition, two 16-bit index registers facilitate program processing of tables and arrays.
- There are three modes of high speed interrupt processing: 8080 compatible, non-Z80 peripheral device, and Z80 Family peripheral with or without daisy chain.
- On-chip dynamic memory refresh counter.

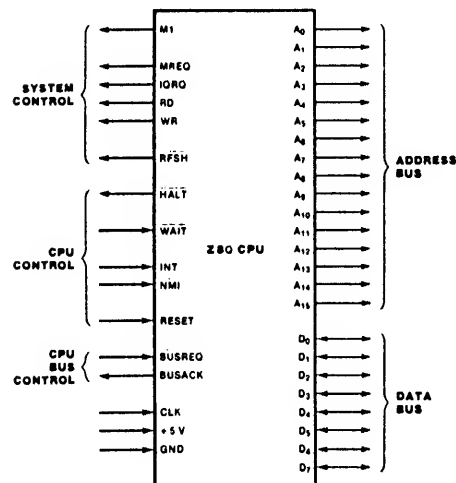


Figure 1. Pin Functions

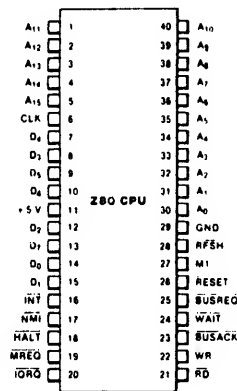


Figure 2. Pin Assignments

2001.0210.0211

ZILOG DATA
Z80 CPU

**General
Description**

The Z80, Z80A, and Z80B CPUs are third-generation single-chip microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second- and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either set of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may

be reserved for very fast interrupt response.

The Z80 also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register. The CPU is easy to incorporate into a system since it requires only a single +5 V power source, all output signals are fully decoded and timed to control standard memory or peripheral circuits, and is supported by an extensive family of peripheral controllers. The internal block diagram (Figure 3) shows the primary functions of the Z80 processors. Subsequent text provides more detail on the Z80 I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.

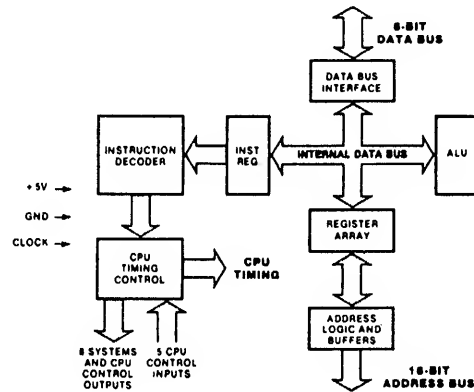


Figure 3. Z80 CPU Block Diagram

Z80 Microprocessor Family

The Zilog Z80 microprocessor is the central element of a comprehensive microprocessor product family. This family works together in most applications with minimum requirements for additional logic, facilitating the design of efficient and cost-effective microcomputer-based systems.

Zilog has designed five components to provide extensive support for the Z80 microprocessor. These are:

- The PIO (Parallel Input/Output) operates in both data-byte I/O transfer mode (with handshaking) and in bit mode (without handshaking). The PIO may be configured to interface with standard parallel peripheral devices such as printers, tape punches, and keyboards.
- The CTC (Counter/Timer Circuit) features four programmable 8-bit counter/timers,

each of which has an 8-bit prescaler. Each of the four channels may be configured to operate in either counter or timer mode.

- The DMA (Direct Memory Access) controller provides dual port data transfer operations and the ability to terminate data transfer as a result of a pattern match.
- The SIO (Serial Input/Output) controller offers two channels. It is capable of operating in a variety of programmable modes for both synchronous and asynchronous communication, including Bi-Synch and SDLC.
- The DART (Dual Asynchronous Receiver/Transmitter) device provides low cost asynchronous serial communication. It has two channels and a full modem control interface.

Z80 CPU Registers

Figure 4 shows three groups of registers within the Z80 CPU. The first group consists of duplicate sets of 8-bit registers: a principal set and an alternate set (designated by ' [prime], e.g., A'). Both sets consist of the Accumulator Register, the Flag Register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques as background-

foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt Register), the R (Refresh Register), the IX and IY (Index Registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus an additional pair of flip-flops which assists in identifying the interrupt mode at any particular time. Table 1 provides further information on these registers.

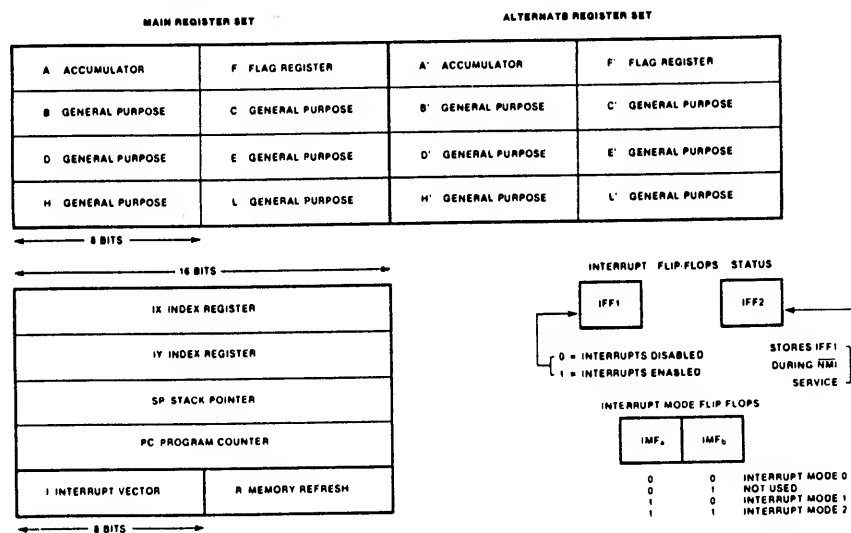


Figure 4. CPU Registers

Z80 CPU Registers (Continued)

Register	Size (Bits)	Remarks
A, A'	8	Stores an operand or the results of an operation.
F, F'	8	See Instruction Set.
B, B'	8	Can be used separately or as a 16-bit register with C.
C, C'	8	See B, above.
D, D'	8	Can be used separately or as a 16-bit register with E.
E, E'	8	See D, above.
H, H'	8	Can be used separately or as a 16-bit register with L.
L, L'	8	See H, above.
Note: The (B,C), (D,E), and (H,L) sets are combined as follows: B — High byte C — Low byte D — High byte E — Low byte H — High byte L — Low byte		
I	8	Stores upper eight bits of memory address for vectored interrupt processing.
R	8	Provides user-transparent dynamic memory refresh. Automatically incremented and placed on the address bus during each instruction fetch cycle.
IX	16	Used for indexed addressing.
IY	16	Same as IX, above.
SP	16	Stores addresses or data temporarily. See Push or Pop in instruction set.
PC	16	Holds address of next instruction.
IFF ₁ -IFF ₂	Flip-Flops	Set or reset to indicate interrupt status (see Figure 4).
IMF _A -IMF _B	Flip-Flops	Reflect Interrupt mode (see Figure 4).

Table 1. Z80 CPU Registers

Interrupts: General Operation

The CPU accepts two interrupt input signals: NMI and INT. The NMI is a non-maskable interrupt and has the highest priority. INT is a lower priority interrupt since it requires that interrupts be enabled in software in order to operate. Either NMI or INT can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80 has a single response mode for interrupt service for the non-maskable interrupt. The maskable interrupt, INT, has three programmable response modes available. These are:

- Mode 0 — compatible with the 8080 micro-processor.

- Mode 1 — Peripheral Interrupt service, for use with non-8080/Z80 systems.
- Mode 2 — a vectored interrupt scheme, usually daisy-chained, for use with Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the NMI and INT signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section.

**Interrupts:
General
Operation**
(Continued)

Non-Maskable Interrupt (NMI). The non-maskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU. NMI is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shut-down after power failure has been detected. After recognition of the NMI signal (providing $\overline{\text{BUSREQ}}$ is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routine.

Maskable Interrupt ($\overline{\text{INT}}$). Regardless of the interrupt mode set by the user, the Z80 response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and $\overline{\text{BUSREQ}}$ is not active) a special interrupt processing cycle begins. This is a special fetch ($\overline{\text{M1}}$) cycle in which $\overline{\text{IORQ}}$ becomes active rather than $\overline{\text{MREQ}}$, as in a normal $\overline{\text{M1}}$ cycle. In addition, this special $\overline{\text{M1}}$ cycle is automatically extended by two $\overline{\text{WAIT}}$ states, to allow for the time required to acknowledge the interrupt request and to place the interrupt vector on the bus.

Mode 0 Interrupt Operation. This mode is compatible with the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus, which is then acted on six times by the CPU. This is normally a Restart Instruction, which will initiate an unconditional jump to the selected one of eight restart locations in page zero of memory.

Mode 1 Interrupt Operation. Mode 1 operation is very similar to that for the NMI. The principal difference is that the Mode 1 interrupt has a vector address of 0038H only.

Mode 2 Interrupt Operation. This interrupt mode has been designed to utilize most effectively the capabilities of the Z80 microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit address vector on the data bus during the interrupt acknowledge cycle. The high-order byte of the interrupt service routine address is supplied by the I (Interrupt) register. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines. These routines may be located at any available

location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

Interrupt Priority (Daisy Chaining and Nested Interrupts). The interrupt priority of each peripheral device is determined by its physical location within a daisy-chain configuration. Each device in the chain has an interrupt enable input line (IEI) and an interrupt enable output line (IEO), which is fed to the next lower priority device. The first device in the daisy chain has its IEI input hardwired to a High level. The first device has highest priority, while each succeeding device has a corresponding lower priority. This arrangement permits the CPU to select the highest priority interrupt from several simultaneously interrupting peripherals.

The interrupting device disables its IEO line to the next lower priority peripheral until it has been serviced. After servicing, its IEO line is raised, allowing lower priority peripherals to demand interrupt servicing.

The Z80 CPU will nest (queue) any pending interrupts or interrupts received while a selected peripheral is being serviced.

Interrupt Enable/Disable Operation. Two flip-flops, IFF_1 and IFF_2 , referred to in the register description are used to signal the CPU interrupt status. Operation of the two flip-flops is described in Table 2. For more details, refer to the *Z80 CPU Technical Manual* and *Z80 Assembly Language Manual*.

Action	IFF_1	IFF_2	Comments
CPU Reset	0	0	Maskable interrupt $\overline{\text{INT}}$ disabled
DI instruction execution	0	0	Maskable interrupt $\overline{\text{INT}}$ disabled
EI instruction execution	1	1	Maskable interrupt $\overline{\text{INT}}$ enabled
LD A,I instruction execution	•	•	$\text{IFF}_2 \rightarrow$ Parity flag
LD A,R instruction execution	•	•	$\text{IFF}_2 \rightarrow$ Parity flag
Accept NMI	0	IFF_1	$\text{IFF}_1 \rightarrow \text{IFF}_2$ (Maskable interrupt $\overline{\text{INT}}$ disabled)
RETN instruction execution	IFF_2	•	$\text{IFF}_2 \rightarrow \text{IFF}_1$ at completion of an NMI service routine.

Table 2. State of Flip-Flops

Instruction Set

The Z80 microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast, efficient data transfers within memory or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the Z80 instruction set and shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. The *Z80 CPU Technical Manual* (03-0029-01) and *Assembly Language Programming Manual* (03-0002-01) contain significantly more details for programming use.

The instructions are divided into the following categories:

- ☐ 8-bit loads
- ☐ 16-bit loads
- ☐ Exchanges, block transfers, and searches
- ☐ 8-bit arithmetic and logic operations
- ☐ General-purpose arithmetic and CPU control

- ☐ 16-bit arithmetic operations
- ☐ Rotates and shifts
- ☐ Bit set, reset, and test operations
- ☐ Jumps
- ☐ Calls, returns, and restarts
- ☐ Input and output operations

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include:

- ☐ Immediate
- ☐ Immediate extended
- ☐ Modified page zero
- ☐ Relative
- ☐ Extended
- ☐ Indexed
- ☐ Register
- ☐ Register indirect
- ☐ Implied
- ☐ Bit

8-Bit Load Group

Mnemonic	Symbolic Operation	S	Z	Flags	P/V	N	C	Opcode	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD r, r'	r ← r'	•	•	X	•	X	•	01 r r'		1	1	4	r, r' Reg.
LD r, n	r ← n	•	•	X	•	X	•	00 r 110		2	2	7	000 B 001 C 010 D 011 E 100 H 101 L 111 A
LD r (HL)	r ← (HL)	•	•	X	•	X	•	01 r 110		1	2	7	
LD r, (IX+d)	r ← (IX+d)	•	•	X	•	X	•	11 011 101	DD	3	5	19	
LD r, (IY+d)	r ← (IY+d)	•	•	X	•	X	•	01 r 101		1	2	7	
LD (HL), r	(HL) ← r	•	•	X	•	X	•	11 111 101	FD	3	5	19	
LD (IX+d), r	(IX+d) ← r	•	•	X	•	X	•	01 r 110		1	2	7	
LD (IY+d), r	(IY+d) ← r	•	•	X	•	X	•	11 011 101	DD	3	5	19	
LD (HL), n	(HL) ← n	•	•	X	•	X	•	01 110 r		1	2	7	
LD (IX+d), n	(IX+d) ← n	•	•	X	•	X	•	11 011 101	DD	4	5	19	
LD (IY+d), n	(IY+d) ← n	•	•	X	•	X	•	01 110 r		1	2	7	
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	00 111 010	3A	3	4	13	
LD (BC), A	(BC) ← A	•	•	X	•	X	•	00 000 010	02	1	2	7	
LD (DE), A	(DE) ← A	•	•	X	•	X	•	00 010 010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	00 110 010	32	3	4	13	
LD A, I	A ← I	•	•	X	•	X	•	11 101 101	ED	2	2	9	
LD A, R	A ← R	•	•	X	•	X	•	01 010 111	57	2	2	9	
LD I, A	I ← A	•	•	X	•	X	•	11 101 101	ED	2	2	9	
LD R, A	R ← A	•	•	X	•	X	•	01 000 111	47	2	2	9	

NOTES: r, r' means any of the registers A, B, C, D, E, H, L.
 IFF the content of the interrupt enable flip-flop (IFF) is copied into the P-V flag.
 For an explanation of flag notation and symbols for mnemonic tables, see Symbolic Notation section following tables.

2001-001

16-Bit Load Group

Mnemonic	Symbolic Operation	S	Z	Flags H	P/V	N	C	Opcode 76 543 210 Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD dd, nn	dd ← nn	*	*	X	*	X	*	00 dd0 001	3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	IX ← nn	*	*	X	*	X	*	11 011 101 DD 00 100 001 21	4	4	14	
LD IY, nn	IY ← nn	*	*	X	*	X	*	11 111 101 FD 00 100 001 21	4	4	14	
LD HL, (nn)	H ← (nn+1) L ← (nn)	*	*	X	*	X	*	00 101 010 2A	3	5	16	
LD dd, (nn)	ddH ← (nn+1) ddL ← (nn)	*	*	X	*	X	*	11 101 101 ED 01 dd1 011	4	6	20	
LD IX, (nn)	IXH ← (nn+1) IXL ← (nn)	*	*	X	*	X	*	11 011 101 DD 00 101 010 2A	4	6	20	
LD IY, (nn)	IYH ← (nn+1) IYL ← (nn)	*	*	X	*	X	*	11 111 101 FD 00 101 010 2A	4	6	20	
LD (nn), HL	(nn+1) ← H (nn) ← L	*	*	X	*	X	*	00 100 010 22	3	5	16	
LD (nn), dd	(nn+1) ← ddH (nn) ← ddL	*	*	X	*	X	*	11 101 101 ED 01 dd0 011	4	6	20	
LD (nn), IX	(nn+1) ← IXH (nn) ← IXL	*	*	X	*	X	*	11 011 101 DD 00 100 010 22	4	6	20	
LD (nn), IY	(nn+1) ← IYH (nn) ← IYL	*	*	X	*	X	*	11 111 101 FD 00 100 010 22	4	6	20	
LD SP, HL	SP ← HL	*	*	X	*	X	*	11 111 001 F9	1	1	6	
LD SP, IX	SP ← IX	*	*	X	*	X	*	11 011 101 DD 11 111 001 F9	2	2	10	
LD SP, IY	SP ← IY	*	*	X	*	X	*	11 111 101 FD 11 111 001 F9	2	2	10	
PUSH qq	(SP-2) ← qqL (SP-1) ← qqH SP ← SP-2	*	*	X	*	X	*	11 qq0 101	1	3	11	qq Pair 00 BC 01 DE 10 HL 11 AF
PUSH IX	(SP-2) ← IXL (SP-1) ← IXH SP ← SP-2	*	*	X	*	X	*	11 011 101 DD 11 100 101 E5	2	4	15	
PUSH IY	(SP-2) ← IYL (SP-1) ← IYH SP ← SP-2	*	*	X	*	X	*	11 111 101 FD 11 100 101 E5	2	4	15	
POP qq	qqH ← (SP+1) qqL ← (SP) SP ← SP+2	*	*	X	*	X	*	11 qq0 001	1	3	10	
POP IX	IXH ← (SP+1) IXL ← (SP) SP ← SP+2	*	*	X	*	X	*	11 011 101 DD 11 100 001 E1	2	4	14	
POP IY	IYH ← (SP+1) IYL ← (SP) SP ← SP+2	*	*	X	*	X	*	11 111 101 FD 11 100 001 E1	2	4	14	

NOTES dd is any of the register pairs BC, DE, HL, SP.
qq is any of the register pairs AF, BC, DE, HL.
(PAIR)_H, (PAIR)_L refer to high order and low order eight bits of the register pair respectively.
* q BCL = C, AFH = A

Exchange, Block Transfer, Block Search Groups

EX DE, HL	DE ← HL	*	*	X	*	X	*	11 101 011 EB	1	1	4	Register bank and auxiliary register bank exchange
EX AF, AF	AF ← AF	*	*	X	*	X	*	00 001 000 08	1	1	4	
EXX	BC ← BC DE ← DE HL ← HL	*	*	X	*	X	*	11 011 001 D9	1	1	4	
EX (SP), HL	H ← (SP+1) L ← (SP)	*	*	X	*	X	*	11 100 011 E3	1	5	19	
EX (SP), IX	IXH ← (SP+1) IXL ← (SP)	*	*	X	*	X	*	11 011 101 DD 11 100 011 E3	2	6	23	
EX (SP), IY	IYH ← (SP+1) IYL ← (SP)	*	*	X	*	X	*	11 111 101 FD 11 100 011 E3	2	6	23	
LDI	(DE) ← (HL) DE ← DE+1 HL ← HL+1 BC ← BC-1	*	*	X	0	X	1	11 101 101 ED 10 100 000 A0	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE) ← (HL) DE ← DE+1 HL ← HL+1 BC ← BC-1 Repeat until BC = 0	*	*	X	0	X	0	11 101 101 ED 10 110 000 B0	2	5	21	1/2 BC = 0 11 BC = 0

NOTE ① P V flag is 0 if the result of BC-1 = 0 otherwise P V = 1

**Exchange,
Block
Transfer,
Block Search
Groups
(Continued)**

Mnemonic	Symbolic Operation	S	Z	Flags H	P/V	N	C	Opcode 78 543 210 Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LDD	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1	•	•	X	0	X	1	0 • 11 101 101 ED 10 101 000 A8	2	4	16	
LDDR	(DE) ← (HL) DE ← DE - 1 HL ← HL - 1 BC ← BC - 1 Repeat until BC = 0	•	•	X	0	X	0	0 • 11 101 101 ED 10 111 000 B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A ← (HL) HL ← HL + 1 BC ← BC - 1	1	1	X	1	X	1	• 11 101 101 ED 10 100 001 A1	2	4	16	
CPIR	A ← (HL) HL ← HL + 1 BC ← BC - 1 Repeat until A = (HL) or BC = 0	1	1	X	1	X	1	• 11 101 101 ED 10 110 001 B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A ← (HL) HL ← HL - 1 BC ← BC - 1	1	1	X	1	X	1	• 11 101 101 ED 10 101 001 A9	2	4	16	
CPDR	A ← (HL) HL ← HL - 1 BC ← BC - 1 Repeat until A = (HL) or BC = 0	1	1	X	1	X	1	• 11 101 101 ED 10 111 001 B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

NOTES ① P/V flag is 0 if the result of BC - 1 = 0 otherwise P/V = 1
② Z flag is 1 if A = (HL) otherwise Z = 0

**8-Bit
Arithmetic
and Logical
Group**

ADD A, r	A ← A + r	1	1	X	1	X	V	0	1	10 000 r	1	1	4	r Reg
ADD A, n	A ← A + n	1	1	X	1	X	V	0	1	11 000 110 — n —	2	2	7	000 B 001 C 010 D 011 E 100 H 101 L 111 A
ADD A, (HL)	A ← A + (HL)	1	1	X	1	X	V	0	1	10 000 110	1	2	7	
ADD A, (IX + d)	A ← A + (IX + d)	1	1	X	1	X	V	0	1	11 011 101 DD 10 000 110 — d —	3	5	19	
ADD A, (IY + d)	A ← A + (IY + d)	1	1	X	1	X	V	0	1	11 111 101 FD 10 000 110 — d —	3	5	19	
ADC A, s	A ← A + s + CY	1	1	X	1	X	V	0	1	0001				s is any of r, n (HL), (IX + d), (IY + d) as shown for ADD instruction. The indicated bits replace the 000 in the ADD set above
SUB s	A ← A - s	1	1	X	1	X	V	1	1	010				
SBC A, s	A ← A - s - CY	1	1	X	1	X	V	1	1	010				
AND s	A ← A ∧ s	1	1	X	1	X	P	0	0	100				
OR s	A ← A ∨ s	1	1	X	0	X	P	0	0	100				
XOR s	A ← A ⊕ s	1	1	X	0	X	P	0	0	100				
CP s	A ← s	1	1	X	1	X	V	1	1	000				
INC r	r ← r + 1	1	1	X	1	X	V	0	•	00 r 100	1	1	4	
INC (HL)	(HL) ← (HL) + 1	1	1	X	1	X	V	0	•	00 110 100	1	3	11	
INC (IX + d)	(IX + d) ← (IX + d) + 1	1	1	X	1	X	V	0	•	11 011 101 DD 00 110 100 — d —	3	6	23	
INC (IY + d)	(IY + d) ← (IY + d) + 1	1	1	X	1	X	V	0	•	11 111 101 FD 00 110 100 — d —	3	6	23	
DEC m	m ← m - 1	1	1	X	1	X	V	1	•	101				m is any of r, (HL), (IX + d), (IY + d) as shown for INC. DEC same format and states as INC. Replace 100 with 101 in opcode

200-1, 001

General-Purpose Arithmetic and CPU Control Groups

Mnemonic	Symbolic Operation	S	Z	Flags H	P/V	N	C	Opcode 78 543 210 Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
DAA	Converts acc. content into packed BCD following add or subtract with packed BCD operands	1	1	X	1	X	P	00 100 111 27	1	1	4	Decimal adjust accumulator
CPL	$A \leftarrow \bar{A}$	*	*	X	1	X	*	00 101 111 2F	1	1	4	Complement accumulator (one's complement)
NEG	$A \leftarrow 0 - A$	1	1	X	1	X	V	11 101 101 ED 01 000 100 44	2	2	8	Negate acc. (two's complement)
CCF	$CY \leftarrow \bar{CY}$	*	*	X	X	X	*	00 111 111 3F	1	1	4	Complement carry flag
SCF	$CY \leftarrow 1$	*	*	X	0	X	*	00 110 111 37	1	1	4	Set carry flag
NOP	No operation	*	*	X	*	X	*	00 000 000 00	1	1	4	
HALT	CPU halted	*	*	X	*	X	*	01 110 110 76	1	1	4	
DI *	IFF = 0	*	*	X	*	X	*	11 110 011 F3	1	1	4	
EI *	IFF = 1	*	*	X	*	X	*	11 111 011 FB	1	1	4	
IM 0	Set interrupt mode 0	*	*	X	*	X	*	11 101 101 ED 01 000 110 46	2	2	8	
IM 1	Set interrupt mode 1	*	*	X	*	X	*	11 101 101 ED 01 010 110 56	2	2	8	
IM 2	Set interrupt mode 2	*	*	X	*	X	*	11 101 101 ED 01 011 110 5E	2	2	8	

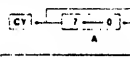
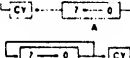
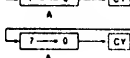

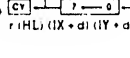
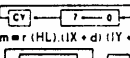
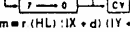
NOTES: IFF indicates the interrupt enable flip flop.
CY indicates the carry flip flop.
* indicates interrupts are not sampled at the end of EI or DI.

16-Bit Arithmetic Group

ADD HL, ss	$HL \leftarrow HL + ss$	*	*	X	X	X	*	00 ss1 001	1	3	11	ss Reg 00 BC 01 DE 10 HL 11 SP
ADC HL, ss	$HL \leftarrow HL + ss + CY$	1	1	X	X	X	V	11 101 101 ED 01 ss1 010	2	4	15	
SBC HL, ss	$HL \leftarrow HL - ss - CY$	1	1	X	X	X	V	11 101 101 ED 01 ss0 010	2	4	15	
ADD IX, pp	$IX \leftarrow IX + pp$	*	*	X	X	X	*	11 pp1 011 DD 01 pp1 001	2	4	15	pp Reg 00 BC 01 DE 10 IX 11 SP
ADD IY, rr	$IY \leftarrow IY + rr$	*	*	X	X	X	*	11 rr1 101 FD 00 rr1 001	2	4	15	rr Reg 00 BC 01 DE 10 IY 11 SP
INC ss	$ss \leftarrow ss + 1$	*	*	X	*	X	*	00 ss0 011	1	1	6	
INC IX	$IX \leftarrow IX + 1$	*	*	X	*	X	*	11 011 101 DD 00 100 011 23	2	2	10	
INC IY	$IY \leftarrow IY + 1$	*	*	X	*	X	*	11 111 101 FD 00 100 011 23	2	2	10	
DEC ss	$ss \leftarrow ss - 1$	*	*	X	*	X	*	00 ss1 011	1	1	6	
DEC IX	$IX \leftarrow IX - 1$	*	*	X	*	X	*	11 011 101 DD 00 101 011 2B	2	2	10	
DEC IY	$IY \leftarrow IY - 1$	*	*	X	*	X	*	11 111 101 FD 00 101 011 2B	2	2	10	

NOTES: ss = any of the register pairs BC, DE, HL, SP.
IX = any of the register pairs BC, DE, IX, SP.
rr = any of the register pairs BC, DE, IY, SP.

Rotate and Shift Group

RLCA		*	*	X	0	X	*	00 000 111 07	1	1	4	Rotate left circular accumulator
RLA		*	*	X	0	X	*	00 010 111 17	1	1	4	Rotate left accumulator
RRCA		*	*	X	0	X	*	00 001 111 0F	1	1	4	Rotate right circular accumulator
RRA		*	*	X	0	X	*	00 011 111 1F	1	1	4	Rotate right accumulator
RLC r		1	1	X	0	X	P	11 001 011 CB 00 000 r	2	2	8	Rotate left circular register r
RLC (HL)		1	1	X	0	X	P	11 001 011 CB 00 000 110	2	4	15	r Reg 000 B 001 C 010 D 011 E 100 H 101 L 111 A
RLC (IX + d)		1	1	X	0	X	P	11 011 101 DD 11 001 011 CB - d - 00 000 110	4	6	23	
RLC (IY + d)		1	1	X	0	X	P	11 111 101 FD 11 001 011 CB - d - 00 000 110	4	6	23	
RL m		1	1	X	0	X	P	010				Instruction format and states are as shown for RLC's. To form new opcode replace 000 or RLC's with shown code.
RRC m		1	1	X	0	X	P	001				

2001 001

Rotate and Shift Group (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags H	P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments	
RR m	 $m \leftarrow r(HL) : (X + d) : (Y + d)$	1	1	X	0	X	P	0	1	017				
SLA m	 $m \leftarrow r(HL) : (X + d) : (Y + d)$	1	1	X	0	X	P	0	1	100				
SRA m	 $m \leftarrow r(HL) : (X + d) : (Y + d)$	1	1	X	0	X	P	0	1	011				
SRL m	 $m \leftarrow r(HL) : (X + d) : (Y + d)$	1	1	X	0	X	P	0	1	111				
PLD	 A	1	1	X	0	X	P	0	*	11 101 101 01 101 111	ED 6F	2	5	18 Rotate digit left and right between the accumulator and location (HL). The content of the upper half of the accumulator is unaffected.
HRD	 A	1	1	X	0	X	P	0	*	11 101 101 01 100 111	ED 67	2	5	18

Bit Set, Reset and Test Group

BIT b, r	$Z \leftarrow r_b$	X	1	X	1	X	X	0	*	11 001 011 01 b r	CB	2	2	8	r Reg 000 B 001 C 010 D 011 E 100 H 101 L 111 A
BIT b, (HL)	$Z \leftarrow (HL)_b$	X	1	X	1	X	X	0	*	11 001 011 01 b 110	CB	2	3	12	b Bit Tested 000 0 001 1 010 2 011 3 100 4 101 5 110 6 111 7
BIT b, (IX + d) _b	$Z \leftarrow (IX + d)_b$	X	1	X	1	X	X	0	*	11 011 101 11 001 011 - d - 01 b 110	DD CB	4	5	20	
BIT b, (IY + d) _b	$Z \leftarrow (IY + d)_b$	X	1	X	1	X	X	0	*	11 111 101 11 001 011 - d - 01 b 110	FD CB	4	5	20	
SET o, r	$r_b \leftarrow 1$	*	*	X	*	X	*	*	*	11 001 011 11 b r	CB	2	2	8	
SET b, (HL)	$(HL)_b \leftarrow 1$	*	*	X	*	X	*	*	*	11 001 011 11 b 110	CB	2	4	15	
SET b, (IX + d)	$(IX + d)_b \leftarrow 1$	*	*	X	*	X	*	*	*	11 011 101 11 001 011 - d - 11 b 110	DD CB	4	6	23	
SET b, (IY + d)	$(IY + d)_b \leftarrow 1$	*	*	X	*	X	*	*	*	11 111 101 11 001 011 - d - 11 b 110	FD CB	4	6	23	
RES b, m	$m_b \leftarrow 0$ $m \leftarrow r(HL) : (IX + d) : (IY + d)$	*	*	X	*	X	*	*	*	11 001 011 11 b 110	CB				To form new opcode replace 11 of SET b, s with 10. Flags and time states for SET instruction.

NOTES: The notation m_b indicates bit b (0 to 7) or location m.

Jump Group

JP nn	$PC \leftarrow nn$	*	*	X	*	X	*	*	*	11 000 011 - n - - n -	C3	3	3	10	
JP cc, nn	If condition cc is true $PC \leftarrow nn$, otherwise continue	*	*	X	*	X	*	*	*	11 cc 010 - n - - n -		3	3	10	cc Condition 000 NZ non-zero 001 Z zero 010 NC non-carry 011 C carry 100 PO parity odd 101 PE parity even 110 P sign positive 111 M sign negative
JR e	$PC \leftarrow PC + e$	*	*	X	*	X	*	*	*	00 011 000 - e - 2 - - e - 2 -	18	2	3	12	
JR C, e	If C = 0, continue If C = 1, $PC \leftarrow PC + e$	*	*	X	*	X	*	*	*	00 111 000 - e - 2 - - e - 2 -	38	2	2	7	If condition not met
JR NC, e	If C = 1, continue If C = 0, $PC \leftarrow PC + e$	*	*	X	*	X	*	*	*	00 110 000 - e - 2 - - e - 2 -	30	2	2	7	If condition not met
JP Z, e	If Z = 0, continue If Z = 1, $PC \leftarrow PC + e$	*	*	X	*	X	*	*	*	00 101 000 - e - 2 - - e - 2 -	28	2	2	7	If condition not met
JR NZ, e	If Z = 1, continue If Z = 0, $PC \leftarrow PC + e$	*	*	X	*	X	*	*	*	00 100 000 - e - 2 - - e - 2 -	20	2	2	7	If condition not met
JP (HL)	$PC \leftarrow HL$	*	*	X	*	X	*	*	*	11 101 001 11 101 001	E9	1	1	4	
JP (IX)	$PC \leftarrow IX$	*	*	X	*	X	*	*	*	11 011 101 11 101 001	DD E9	2	2	8	

2001-001

Jump Group (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags H	P/V	N	C	Opcode 78 543 210 Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
IP (IY)	PC ← IY	•	•	X	•	X	•	11 111 101 FD	2	2	8	
D/NZ, •	B ← B - 1 If B = 0, continue If B ≠ 0, PC ← PC + •	•	•	X	•	X	•	11 101 001 E9 00 010 000 10 — • - 2 —	2	2	6	If B = 0.
									2	3	13	If B ≠ 0.

NOTES • represents the extension in the relative addressing mode
• is a signed two's complement number in the range < -126, 126 >
• - 2 in the opcode provides an effective address of PC + • as PC is incremented by 2 prior to the addition of •.

Call and Return Group

CALL nn	(SP - 1) ← PC _H (SP - 2) ← PC _L PC ← nn	•	•	X	•	X	•	11 001 101 CD — n — — n —	3	5	17	
CALL cc, nn	If condition cc is false, continue, otherwise same as CALL nn	•	•	X	•	X	•	11 cc 100 — n — — n —	3	3	10	If cc is false.
									3	5	17	If cc is true.
RET	PC _L ← (SP) PC _H ← (SP + 1)	•	•	X	•	X	•	11 001 001 C9	1	3	10	
RET cc	If condition cc is false, continue, otherwise same as RET	•	•	X	•	X	•	11 cc 000	1	1	5	If cc is false
									1	3	11	If cc is true
RETI	Return from interrupt	•	•	X	•	X	•	11 101 101 ED 01 001 101 4D	2	4	14	cc Condition 000 NZ non-zero 001 Z zero 010 NC non-carry 011 C carry 100 PO parity odd 101 PE parity even 110 P sign positive 111 M sign negative
RETI	Return from non-maskable interrupt	•	•	X	•	X	•	01 000 101 45	2	4	14	
RST p	(SP - 1) ← PC _H (SP - 2) ← PC _L PC _H ← 0 PC _L ← p	•	•	X	•	X	•	11 1 111	1	3	11	1 p 000 00H 001 08H 010 10H 011 18H 100 20H 101 28H 110 30H 111 38H

NOTE • RETN loads IFF₂ ← IFF₁

Input and Output Group

IN A, (n)	A ← (n)	•	•	X	•	X	•	11 011 011 DB — n — — n —	2	3	11	n to A ₀ - A ₇ Acc. to A ₈ - A ₁₅
IN r, (C)	r ← (C) If r = 110 only the flag will be selected	1	1	X	1	X	P 0 •	11 101 101 ED 01 r 000	2	3	12	C to A ₀ - A ₇ B to A ₈ - A ₁₅
INI	(HL) ← (C) B ← B - 1 HL ← HL + 1	X	1	X	X	X	X 1 •	11 101 101 ED 10 100 010 A2	2	4	16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
INIR	(HL) ← (C) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X 1 •	11 101 101 ED 10 110 010 B2	2 2 (If B = 0)	5 4 (If B = 0)	21 16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
IND	(HL) ← (C) B ← B - 1 HL ← HL - 1	X	1	X	X	X	X 1 •	11 101 101 ED 10 101 010 AA	2	4	16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
INDR	(HL) ← (C) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X 1 •	11 101 101 ED 10 111 010 BA	2 2 (If B = 0)	5 4 (If B = 0)	21 16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
OUT (n), A	(n) ← A	•	•	X	•	X	•	11 010 011 D3 — n — — n —	2	3	11	n to A ₀ - A ₇ Acc. to A ₈ - A ₁₅
OUT (C), r	(C) ← r	•	•	X	•	X	•	11 101 101 ED 01 r 001	2	3	12	C to A ₀ - A ₇ B to A ₈ - A ₁₅
OUTI	(C) ← (HL) B ← B - 1 HL ← HL + 1	X	1	X	X	X	X 1 •	11 101 101 ED 10 100 011 A3	2	4	16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
OTIR	(C) ← (HL) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X 1 •	11 101 101 ED 10 110 011 B3	2 2 (If B = 0)	5 4 (If B = 0)	21 16	C to A ₀ - A ₇ B to A ₈ - A ₁₅
OUTD	(C) ← (HL) B ← B - 1 HL ← HL - 1	X	1	X	X	X	X 1 •	11 101 101 ED 10 101 011 AB	2	4	16	C to A ₀ - A ₇ B to A ₈ - A ₁₅

NOTE ① If the result of B - 1 is zero the Z flag is set, otherwise it is reset

2001-001

ZILOG DATA
Z80 CPU

Input and Output Group (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags H P/V N C	Opcode 76 543 210 Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
OTDR	(C) - (HL) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X X X X X 1 *	11 101 101 ED 10 111 011	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to A ₀ - A ₇ B to A ₈ - A ₁₅

Summary of Flag Operation

Instruction	D ₇ S	Z	H	P/V	N	D ₀ C	Comments
ADD A, s; ADC A, s	1	1	X	1	X	V 0 1	8-bit add or add with carry
SUB s; SBC A, s; CP s; NEG	1	1	X	1	X	V 1 1	8-bit subtract, subtract with carry, compare and negate accumulator.
AND s	1	1	X	1	X	P 0 0	Logical operations
OR s; XOR s	1	1	X	0	X	P 0 0	
INC s	1	1	X	1	X	V 0 *	8-bit increment
DEC s	1	1	X	1	X	V 1 *	8-bit decrement
ADD DD, ss	*	*	X	X	X	* 0 1	16-bit add
ADC HL, ss	1	1	X	X	X	V 0 1	16-bit add with carry
SBC HL, ss	1	1	X	X	X	V 1 1	16-bit subtract with carry
RLA; RLCA; RRA; RRCA	*	*	X	0	X	* 0 1	Rotate accumulator
RL m; RLC m; RR m;	1	1	X	0	X	P 0 1	
RRC m; SLA m;							Rotate and shift locations.
SRA m; SRL m							
RLD; RRD	1	1	X	0	X	P 0 *	Rotate digit left and right
DAA	1	1	X	1	X	P * 1	Decimal adjust accumulator
CPL	*	*	X	1	X	* 1 *	Complement accumulator
SCF	*	*	X	0	X	* 0 1	Set carry
CCF	*	*	X	X	X	* 0 1	Complement carry
IN r (C)	1	1	X	0	X	P 0 *	Input register indirect
OUT r (C)	X	1	X	X	X	X 1 *	
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X 1 *	Block input and output Z = 0 if B = 0 otherwise Z = 0
LDIR; LDDR	X	X	X	0	X	1 0 *	
CPI; CPDR; CPD; CPDR	X	1	X	X	X	1 *	Block search instructions Z = 1 if A = (HL), otherwise Z = 0 P/V = 1 if BC ≠ 0 otherwise P/V = 0
LD A, i; LD A, R	1	1	X	0	X	IFF 0 *	
BIT b, s	X	1	X	1	X	X 0 *	The state of bit b of location s is copied into the Z flag.

Symbolic Notation

Symbol	Operation	Symbol	Operation
S	Sign flag. S = 1 if the MSB of the result is 1.	I	The flag is affected according to the result of the operation.
Z	Zero flag. Z = 1 if the result of the operation is 0.	*	The flag is unchanged by the operation.
P/V	Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V = 1 if the result of the operation is even, P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow.	0	The flag is reset by the operation.
H	Half-carry flag. H = 1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator.	1	The flag is set by the operation.
N	Add/Subtract flag. N = 1 if the previous operation was a subtract.	X	The flag is a "don't care."
H & N	H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.	V	P/V flag affected according to the overflow result of the operation.
C	Carry/Link flag. C = 1 if the operation produced a carry from the MSB of the operand or result.	P	P/V flag affected according to the parity result of the operation.
		r	Any one of the CPU registers A, B, C, D, E, H, L.
		s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
		ss	Any 16-bit location for all the addressing modes allowed for that instruction.
		ii	Any one of the two index registers IX or IY.
		R	Refresh counter.
		n	8-bit value in range < 0, 255 >.
		nn	16-bit value in range < 0, 65535 >.

2001-001

Pin	Descriptions
A₀-A₁₅	Address Bus (output, active High, 3-state). A ₀ -A ₁₅ form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.
BUSACK	Bus Acknowledge (output, active Low). Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals MREQ , IORQ , RD , and WR have entered their high-impedance states. The external circuitry can now control these lines.
BUSREQ	Bus Request (input, active Low). Bus Request has a higher priority than NMI and is always recognized at the end of the current machine cycle. BUSREQ forces the CPU address bus, data bus, and control signals MREQ , IORQ , RD , and WR to go to a high-impedance state so that other devices can control these lines. BUSREQ is normally wire-ORed and requires an external pullup for these applications. Extended BUSREQ periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.
D₀-D₇	Data Bus (input/output, active High, 3-state). D ₀ -D ₇ constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.
HALT	Halt State (output, active Low). HALT indicates that the CPU has executed a Halt instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can resume. While halted, the CPU executes NOPs to maintain memory refresh.
INT	Interrupt Request (input, active Low). Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software-controlled interrupt enable flip-flop (IFF) is enabled. INT is normally wire-ORed and requires an external pullup for these applications.
IORQ	Input/Output Request (output, active Low, 3-state). IORQ indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation. IORQ is also generated concurrently with M1 during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.
M1	Machine Cycle One (output, active Low). M1 , together with MREQ , indicates that the current machine cycle is the opcode fetch cycle of an instruction execution. M1 , together with IORQ , indicates an interrupt acknowledge cycle.
MREQ	Memory Request (output, active Low, 3-state). MREQ indicates that the address bus holds a valid address for a memory read or memory write operation.
NMI	Non-Maskable Interrupt (input, active Low). NMI has a higher priority than INT . NMI is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.
RD	Memory Read (output, active Low, 3-state). RD indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.
RESET	Reset (input, active Low). RESET initializes the CPU as follows: it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state. Note that RESET must be active for a minimum of three full clock cycles before the reset operation is complete.
RFSH	Refresh (output, active Low). RFSH , together with MREQ , indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.
WAIT	Wait (input, active Low). WAIT indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended WAIT periods can prevent the CPU from refreshing dynamic memory properly.
WR	Memory Write (output, active Low, 3-state). WR indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

CPU Timing

The Z80 CPU executes instructions by proceeding through a specific sequence of operations:

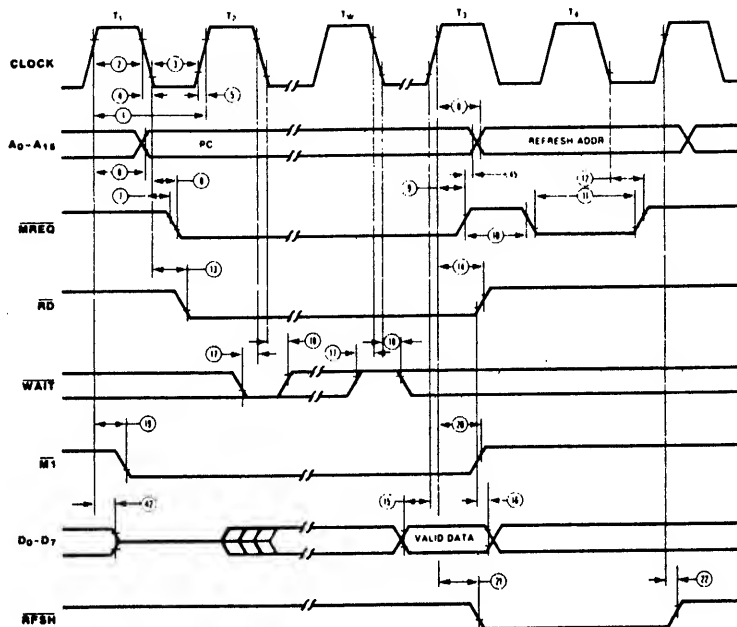
- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a T time or cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

Instruction Opcode Fetch. The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (Figure 5). Approximately one-half clock cycle later, \overline{MREQ} goes active. The falling edge of \overline{MREQ} can be used directly as a Chip Enable to dynamic memories. When active, \overline{RD} indicates that the memory data can be enabled onto the CPU

data bus.

The CPU samples the \overline{WAIT} input with the rising edge of clock state T3. During clock states T3 and T4 of an M1 cycle dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction. When the Refresh Control signal becomes active, refreshing of dynamic memory can take place.



NOTE: T_w - Wait cycle added when necessary for slow ancillary devices.

Figure 5. Instruction Opcode Fetch

**CPU
Timing
(Continued)**

Memory Read or Write Cycles. Figure 6 shows the timing of memory read or write cycles other than an opcode fetch (M1) cycle. The $\overline{\text{MREQ}}$ and RD signals function exactly as in the fetch cycle. In a memory write cycle, $\overline{\text{MREQ}}$ also becomes active when the address

bus is stable, so that it can be used directly as a Chip Enable for dynamic memories. The $\overline{\text{WR}}$ line is active when the data bus is stable, so that it can be used directly as an R/W pulse to most semiconductor memories.

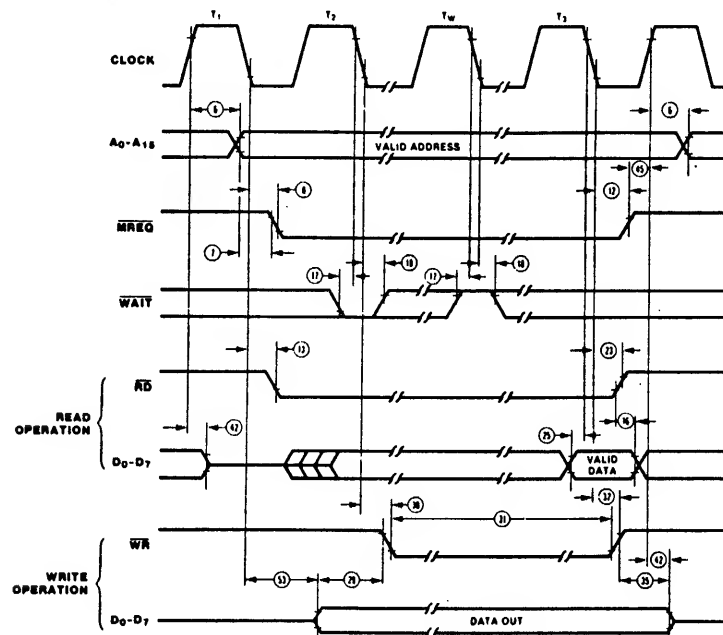
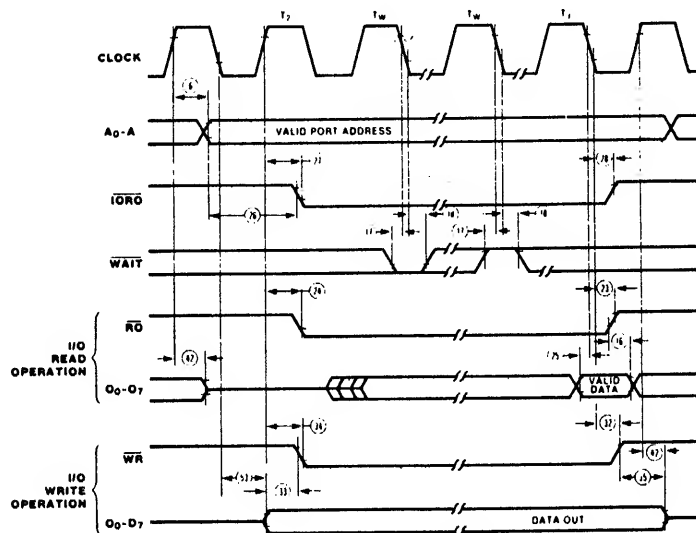


Figure 6. Memory Read or Write Cycles

**CPU
Timing**
(Continued)

Input or Output Cycles. Figure 7 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically

inserts a single Wait state (T_w). This extra Wait state allows sufficient time for an I/O port to decode the address and the port address lines.

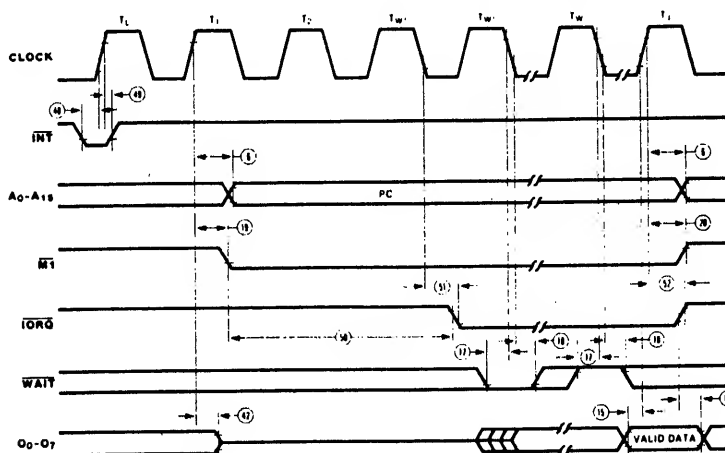


NOTE: T_w = One Wait cycle automatically inserted by CPU.

Figure 7. Input or Output Cycles

Interrupt Request/Acknowledge Cycle. The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 8). When an interrupt is accepted, a special $\overline{M1}$ cycle is generated.

During this $\overline{M1}$ cycle, \overline{IORQ} becomes active (instead of \overline{MREQ}) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.



NOTE: 1) T_L = Last state of previous instruction.

2) Two Wait cycles automatically inserted by CPU(*).

Figure 8. Interrupt Request/Acknowledge Cycle

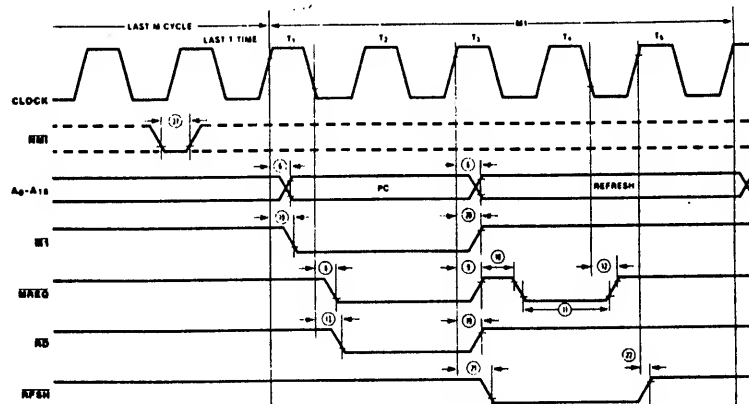
2005-884, 885

ZILOG DATA
Z80 CPU

**CPU
Timing**
(Continued)

Non-Maskable Interrupt Request Cycle. NMI is sampled at the same time as the maskable interrupt input \overline{INT} but has higher priority and cannot be disabled under software control. The subsequent timing is similar to

that of a normal memory read operation except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the NMI service routine located at address 0066H (Figure 9).



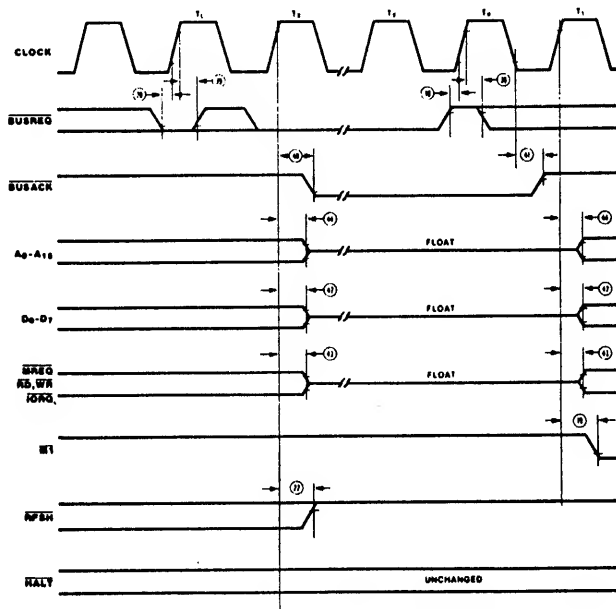
* Although NMI is an asynchronous input, to guarantee its being recognized on the following machine cycle, NMI's falling edge

must occur no later than the rising edge of the clock cycle preceding T_{LAST} .

Figure 9. Non-Maskable Interrupt Request Operation

Bus Request/Acknowledge Cycle. The CPU samples \overline{BUSREQ} with the rising edge of the last clock period of any machine cycle (Figure 10). If \overline{BUSREQ} is active, the CPU sets its address, data, and \overline{MREQ} , \overline{IORQ} , \overline{RD} , and \overline{WR}

lines to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



NOTE: T_L = Last state of any M cycle.

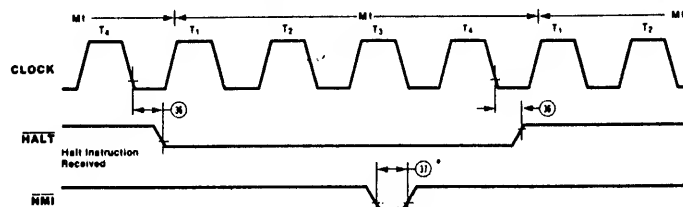
T_X = An arbitrary clock cycle used by requesting device.

Figure 10. Bus Request/Acknowledge Cycle

**CPU
Timing**
(Continued)

Halt Acknowledge Cycle. When the CPU receives a $\overline{\text{HALT}}$ instruction, it executes NOP states until either an $\overline{\text{INT}}$ or $\overline{\text{NMI}}$ input is

received. When in the Halt state, the $\overline{\text{HALT}}$ output is active and remains so until an interrupt is processed (Figure 11).



NOTE: $\overline{\text{INT}}$ will also force a Halt exit.

*See note, Figure 9.

Figure 11. Halt Acknowledge Cycle

Reset Cycle. $\overline{\text{RESET}}$ must be active for at least three clock cycles for the CPU to properly accept it. As long as $\overline{\text{RESET}}$ remains active, the address and data buses float, and the control outputs are inactive. Once $\overline{\text{RESET}}$ goes

inactive, two internal T cycles are consumed before the CPU resumes normal processing operation. $\overline{\text{RESET}}$ clears the PC register, so the first opcode fetch will be to location 0000 (Figure 12).

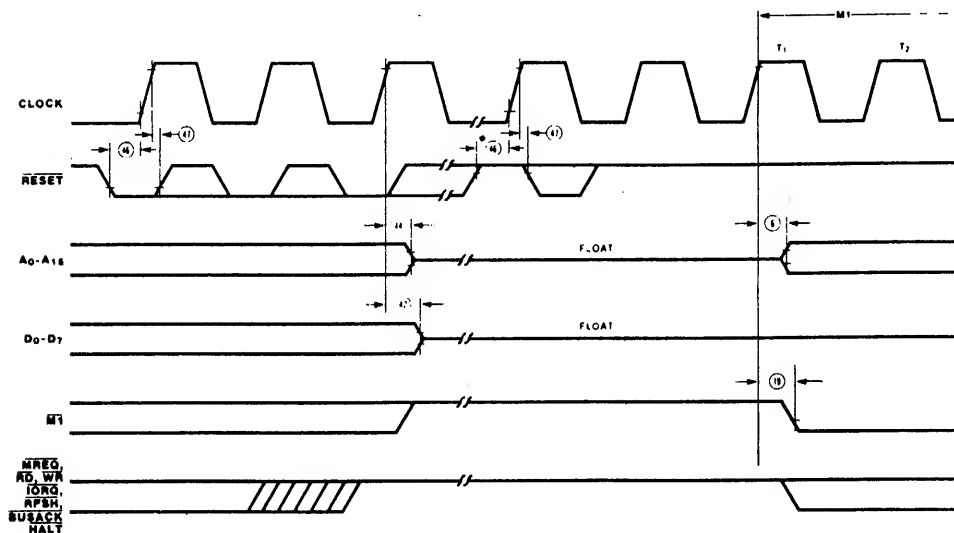


Figure 12. Reset Cycle

AC Charac- teristics	Number	Symbol	Parameter	Z80 CPU		Z80A CPU		Z80B CPU	
				Min (ns)	Max (ns)	Min (ns)	Max (ns)	Min (ns)	Max (ns)
	1	TcC	Clock Cycle Time	400*		250*		165*	
	2	TwCh	Clock Pulse Width (High)	180*		110*		65*	
	3	TwCl	Clock Pulse Width (Low)	180	2000	110	2000	65	2000
	4	TIC	Clock Fall Time	—	30	—	30	—	20
	5	TrC	Clock Rise Time	—	30	—	30	—	20
	6	TdCr(A)	Clock I to Address Valid Delay	—	145	—	110	—	90
	7	TdA(MREQI)	Address Valid to $\overline{\text{MREQ}}$ I Delay	125*	—	65*	—	35*	—
	8	TdCI(MREQI)	Clock I to $\overline{\text{MREQ}}$ I Delay	—	100	—	85	—	70
	9	TdCr(MREQr)	Clock I to $\overline{\text{MREQ}}$ I Delay	—	100	—	85	—	70
	10	TwMREQh	$\overline{\text{MREQ}}$ Pulse Width (High)	170*	—	110*	—	65*	—
	11	TwMREQl	$\overline{\text{MREQ}}$ Pulse Width (Low)	360*	—	220*	—	135*	—
	12	TdCI(MREQr)	Clock I to $\overline{\text{MREQ}}$ I Delay	—	100	—	85	—	70
	13	TdCI(RDI)	Clock I to $\overline{\text{RD}}$ I Delay	—	130	—	95	—	80
	14	TdCr(RDr)	Clock I to $\overline{\text{RD}}$ I Delay	—	100	—	85	—	70
	15	TsD(Cr)	Data Setup Time to Clock I	50	—	35	—	30	—
	16	ThD(RDr)	Data Hold Time to $\overline{\text{RD}}$ I	—	0	—	0	—	0
	17	TsWAIT(CI)	$\overline{\text{WAIT}}$ Setup Time to Clock I	70	—	70	—	60	—
	18	ThWAIT(CI)	$\overline{\text{WAIT}}$ Hold Time after Clock I	—	0	—	0	—	0
	19	TdCr(MIi)	Clock I to $\overline{\text{MI}}$ I Delay	—	130	—	100	—	80
	20	TdCr(MIr)	Clock I to $\overline{\text{MI}}$ I Delay	—	130	—	100	—	80
	21	TdCr(RFSHi)	Clock I to $\overline{\text{RFSH}}$ I Delay	—	180	—	130	—	110
	22	TdCr(RFSHr)	Clock I to $\overline{\text{RFSH}}$ I Delay	—	150	—	120	—	100
	23	TdCI(RDr)	Clock I to $\overline{\text{RD}}$ I Delay	—	110	—	85	—	70
	24	TdCr(RDI)	Clock I to $\overline{\text{RD}}$ I Delay	—	100	—	85	—	70
	25	TsD(CI)	Data Setup to Clock I during $\text{M}_2, \text{M}_3, \text{M}_4$ or M_5 Cycles	60	—	50	—	40	—
	26	TdA(IORQi)	Address Stable prior to $\overline{\text{IORQ}}$ I	320*	—	180*	—	110*	—
	27	TdCr(IORQi)	Clock I to $\overline{\text{IORQ}}$ I Delay	—	90	—	75	—	65
	28	TdCI(IORQr)	Clock I to $\overline{\text{IORQ}}$ I Delay	—	110	—	85	—	70
	29	TdD(WRi)	Data Stable prior to $\overline{\text{WR}}$ I	190*	—	80*	—	25*	—
	30	TdCI(WRi)	Clock I to $\overline{\text{WR}}$ I Delay	—	90	—	80	—	70
	31	TwWR	$\overline{\text{WR}}$ Pulse Width	360*	—	220*	—	135*	—
	32	TdCI(WRr)	Clock I to $\overline{\text{WR}}$ I Delay	—	100	—	80	—	70
	33	TdD(WRi)	Data Stable prior to $\overline{\text{WR}}$ I	20*	—	-10*	—	-55*	—
	34	TdCr(WRi)	Clock I to $\overline{\text{WR}}$ I Delay	—	80	—	65	—	60
	35	TdWRr(D)	Data Stable from $\overline{\text{WR}}$ I	120*	—	60*	—	30*	—
	36	TdCI(HALT)	Clock I to $\overline{\text{HALT}}$ I or I	—	300	—	300	—	260
	37	TwNMI	$\overline{\text{NMI}}$ Pulse Width	80	—	80	—	70	—
	38	TsBUSREQ(Cr)	$\overline{\text{BUSREQ}}$ Setup Time to Clock I	80	—	50	—	50	—

*For clock periods other than the minimums shown in the table, calculate parameters using the expressions in the table on the following page.

AC Charac- teristics (Continued)	Number	Symbol	Parameter	Z80 CPU		Z80A CPU		Z80B CPU	
				Min (ns)	Max (ns)	Min (ns)	Max (ns)	Min (ns)	Max (ns)
	39	ThBUSREQ(Cr)	BUSREQ Hold Time after Clock 1	0	—	0	—	0	—
	40	TdCr(BUSACKI)	Clock 1 to BUSACK 1 Delay	—	120	—	100	—	90
	41	TdC(BUSACKr)	Clock 1 to BUSACK 1 Delay	—	110	—	100	—	90
	42	TdCr(Dz)	Clock 1 to Data Float Delay	—	90	—	90	—	80
	43	TdCr(CTz)	Clock 1 to Control Outputs Float Delay (MREQ, IORQ, RD, and WR)	—	110	—	80	—	70
	44	TdCr(Az)	Clock 1 to Address Float Delay	—	110	—	90	—	80
	45	TdCTr(A)	Address Stable after MREQ 1, IORQ 1, RD 1, and WR 1	160*	—	80*	—	35*	—
	46	TsRESET(Cr)	RESET to Clock 1 Setup Time	90	—	60	—	60	—
	47	ThRESET(Cr)	RESET to Clock 1 Hold Time	—	0	—	0	—	0
	48	TsINT(Cr)	INT to Clock 1 Setup Time	80	—	80	—	70	—
	49	ThINTr(Cr)	INT to Clock 1 Hold Time	—	0	—	0	—	0
	50	TdMII(IORQI)	M 1 to IORQ 1 Delay	920*	—	565*	—	365*	—
	51	TdCI(IORQI)	Clock 1 to IORQ 1 Delay	—	110	—	85	—	70
	52	TdCI(IORQr)	Clock 1 to IORQ 1 Delay	—	100	—	85	—	70
	53	TdCI(D)	Clock 1 to Data Valid Delay	—	230	—	150	—	130

*For clock periods other than the minimums shown in the table, calculate parameters using the following expressions. Calculated values above assumed TrC = TIC = 20 ns

Footnotes to AC Characteristics

Number	Symbol	Z80	Z80A	Z80B
1	TcC	TwCh + TwCI + TrC + TIC	TwCh + TwCI + TrC + TIC	TwCh + TwCI + TrC + TIC
2	TwCh	Although static by design, TwCh of greater than 200 μ s is not guaranteed	Although static by design, TwCh of greater than 200 μ s is not guaranteed	Although static by design, TwCh of greater than 200 μ s is not guaranteed
7	TdA(MREQI)	TwCh + TIC - 75	TwCh + TIC - 65	TwCh + TIC - 50
10	TwMREQh	TwCh + TIC - 30	TwCh + TIC - 20	TwCh + TIC - 20
11	TwMREQI	TcC - 40	TcC - 30	TcC - 30
26	TdA(IORQI)	TcC - 80	TcC - 70	TcC - 55
29	TdD(WRI)	TcC - 210	TcC - 170	TcC - 140
31	TwWR	TcC - 40	TcC - 30	TcC - 30
33	TdD(WRI)	TwCI + TrC - 180	TwCI + TrC - 140	TwCI + TrC - 140
35	TdWRR(D)	TwCI + TrC - 80	TwCI + TrC - 70	TwCI + TrC - 55
45	TdCTr(A)	TwCI + TrC - 40	TwCI + TrC - 50	TwCI + TrC - 50
50	TdMII(IORQI)	2TcC + TwCh + TIC - 80	2TcC + TwCh + TIC - 65	2TcC + TwCh + TIC - 50

AC Test Conditions:
V_{IH} = 2.0 V
V_{IL} = 0.8 V
V_{IHC} = V_{CC} - 0.6 V
V_{ILC} = 0.45 V

V_{OH} = 2.0 V
V_{OL} = 0.8 V
F_{LOAT} = \pm 0.5 V

ZILOG DATA Z80 CPU

Absolute Maximum Ratings

Storage Temperature -65°C to +150°C

Temperature under Bias Specified operating range

Voltages on all inputs and outputs with respect to ground . -0.3 V to +7 V

Power Dissipation 1.5 W

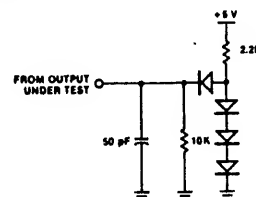
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Standard Test Conditions

The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are:

- 0°C to +70°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -40°C to +85°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -55°C to +125°C,
+4.5 V ≤ V_{CC} ≤ +5.5 V

All ac parameters assume a load capacitance of 50 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 200 pF for the data bus and 100 pF for address and control lines.



DC Characteristics	Symbol	Parameter	Min	Max	Unit	Test Condition
	V _{ILC}	Clock Input Low Voltage	-0.3	0.45	V	
	V _{IHC}	Clock Input High Voltage	V _{CC} - .6	V _{CC} + .3	V	
	V _{IL}	Input Low Voltage	-0.3	0.8	V	
	V _{IH}	Input High Voltage	2.0	V _{CC}	V	
	V _{OL}	Output Low Voltage		0.4	V	I _{OL} = 1.8 mA
	V _{OH}	Output High Voltage	2.4		V	I _{OH} = -250 μA
	I _{CC}	Power Supply Current				
		Z80		150 ¹	mA	
		Z80A		200 ²	mA	
		Z80B		200	mA	
	I _{LI}	Input Leakage Current		10	μA	V _{IN} = 0 to V _{CC}
	I _{LEAK}	3-State Output Leakage Current in Float	-10	10 ³	μA	V _{OUT} = 0.4 to V _{CC}

1. For military grade parts, I_{CC} is 200 mA.
2. Typical rate for Z80A is 90 mA.

3. A15-A0, D7-D0, MREQ, IORQ, RD, and WR

Capacitance	Symbol	Parameter	Min	Max	Unit	Note
	C _{CLOCK}	Clock Capacitance		35	pF	
	C _{IN}	Input Capacitance		5	pF	Unmeasured pins returned to ground
	C _{OUT}	Output Capacitance		10	pF	

T_A = 25°C, f = 1 MHz

Ordering Information	Product Number	Package/Temp	Speed	Description	Product Number	Package/Temp	Speed	Description
	Z8400	CE	2.5 MHz	Z80 CPU (40-pin)	Z8400A	DE	4.0 MHz	Z80A CPU (40-pin)
	Z8400	CM	2.5 MHz	Same as above	Z8400A	DS	4.0 MHz	Same as above
	Z8400	CMB	2.5 MHz	Same as above	Z8400A	PE	4.0 MHz	Same as above
	Z8400	CS	2.5 MHz	Same as above	Z8400A	PS	4.0 MHz	Same as above
	Z8400	DE	2.5 MHz	Same as above	Z8400B	CE	6.0 MHz	Z80B CPU (40-pin)
	Z8400	DS	2.5 MHz	Same as above	Z8400B	CM	6.0 MHz	Same as above
	Z8400	PE	2.5 MHz	Same as above	Z8400B	CMB	6.0 MHz	Same as above
	Z8400	PS	2.5 MHz	Same as above	Z8400B	CS	6.0 MHz	Same as above
	Z8400A	CE	4.0 MHz	Z80A CPU (40 pin)	Z8400B	DE	6.0 MHz	Same as above
	Z8400A	CM	4.0 MHz	Same as above	Z8400B	DS	6.0 MHz	Same as above
	Z8400A	CMB	4.0 MHz	Same as above	Z8400B	PE	6.0 MHz	Same as above
	Z8400A	CS	4.0 MHz	Same as above	Z8400B	PS	6.0 MHz	Same as above

NOTES: C = Ceramic D = Cerdip, P = Plastic; E = -40°C to +85°C, M = -55°C to +125°C, MB = -55°C to +125°C with MIL-STD-883 Class B processing; S = 0°C to +75°C

ZILOG DATA
Z80 CPU

Z8420 Z80[®] PIO Parallel Input/Output Controller



Product Specification

March 1981

- Features**
- Provides a direct interface between Z-80 microcomputer systems and peripheral devices.
 - Both ports have interrupt-driven handshake for fast response.
 - Four programmable operating modes: byte input, byte output, byte input/output (Port A only), and bit input/output.

- Programmable interrupts on peripheral status conditions.
- Standard Z-80 Family bus-request and prioritized interrupt-request daisy chains implemented without external logic.
- The eight Port B outputs can drive Darlington transistors (1.5 mA at 1.5 V).

General Description

The Z-80 PIO Parallel I/O Circuit is a programmable, dual-port device that provides a TTL-compatible interface between peripheral devices and the Z-80 CPU. The CPU configures the Z-80 PIO to interface with a wide range of peripheral devices with no other external logic. Typical peripheral devices that are compatible with the Z-80 PIO include most keyboards, paper tape readers and punches, printers, PROM programmers, etc.

One characteristic of the Z-80 peripheral controllers that separates them from other interface controllers is that all data transfer between the peripheral device and the CPU is

accomplished under interrupt control. Thus, the interrupt logic of the PIO permits full use of the efficient interrupt capabilities of the Z-80 CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO.

Another feature of the PIO is the ability to interrupt the CPU upon occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the time the processor must spend in polling peripheral status.

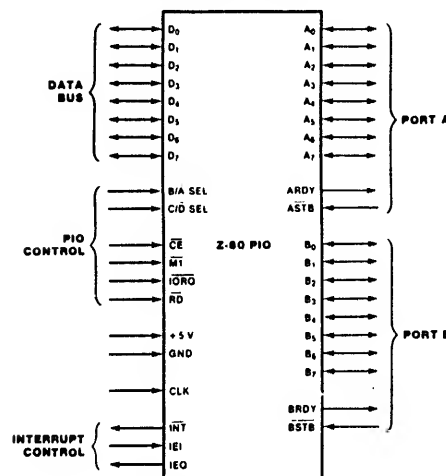


Figure 1. Pin Functions

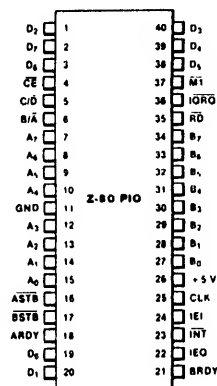


Figure 2. Pin Assignments

2006 0297 0298

ZILOG DATA
Z80 PIO

**General
Description**
(Continued)

The Z-80 PIO interfaces to peripherals via two independent general-purpose I/O ports, designated Port A and Port B. Each port has eight data bits and two handshake signals, Ready and Strobe, which control data transfer. The Ready output indicates to the peripheral that the port is ready for a data transfer. Strobe is an input from the peripheral that indicates when a data transfer has occurred.

Operating Modes. The Z-80 PIO ports can be programmed to operate in four modes: byte output (Mode 0), byte input (Mode 1), byte input/output (Mode 2) and bit input/output (Mode 3).

In Mode 0, either Port A or Port B can be programmed to output data. Both ports have output registers that are individually addressed by the CPU; data can be written to either port at any time. When data is written to a port, an active Ready output indicates to the external device that data is available at the associated port and is ready for transfer to the external device. After the data transfer, the external device responds with an active Strobe input, which generates an interrupt, if enabled.

In Mode 1, either Port A or Port B can be configured in the input mode. Each port has an input register addressed by the CPU. When the CPU reads data from a port, the PIO sets the Ready signal, which is detected by the external device. The external device then places data on the I/O lines and strobes the I/O port, which latches the data into the Port Input Register, resets Ready, and triggers the Interrupt Request, if enabled. The CPU can read the input data at any time, which again sets Ready.

Mode 2 is bidirectional and uses Port A, plus the interrupts and handshake signals from both ports. Port B must be set to Mode 3 and masked off. In operation, Port A is used for both data input and output. Output operation is similar to Mode 0 except that data is allowed out onto the Port A bus only when \overline{ASTB} is Low. For input, operation is similar to Mode 1, except that the data input uses the Port B handshake signals and the Port B interrupt (if enabled).

Both ports can be used in Mode 3. In this mode, the individual bits are defined as either input or output bits. This provides up to eight separate, individually defined bits for each port. During operation, Ready and Strobe are

not used. Instead, an interrupt is generated if the condition of one input changes, or if all inputs change. The requirements for generating an interrupt are defined during the programming operation; the active level is specified as either High or Low, and the logic condition is specified as either one input active (OR) or all inputs active (AND). For example, if the port is programmed for active Low inputs and the logic function is AND, then all inputs at the specified port must go Low to generate an interrupt.

Data outputs are controlled by the CPU and can be written or changed at any time.

- Individual bits can be masked off.
- The handshake signals are not used in Mode 3; Ready is held Low, and Strobe is disabled.
- When using the Z-80 PIO interrupts, the Z-80 CPU interrupt mode must be set to Mode 2.

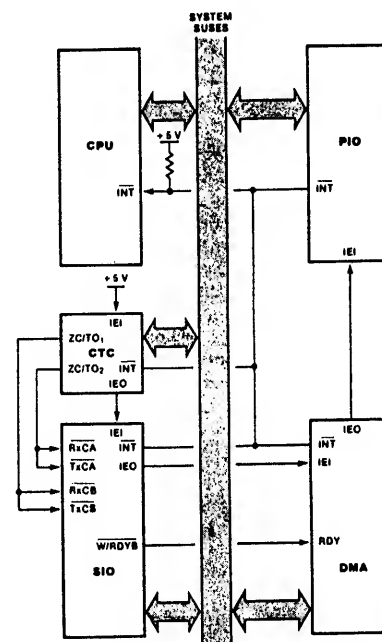


Figure 3. PIO in a Typical Z80 Family Environment

Internal Structure

The internal structure of the Z-80 PIO consists of a Z-80 CPU bus interface, internal control logic, Port A I/O logic, Port B I/O logic, and interrupt control logic (Figure 4). The CPU bus interface logic allows the Z-80 PIO to interface directly to the Z-80 CPU with no other external logic. The internal control logic synchronizes the CPU data bus to the peripheral device interfaces (Port A and Port B). The two I/O ports (A and B) are virtually identical and are used to interface directly to peripheral devices.

Port Logic. Each port contains separate input and output registers, handshake control logic, and the control registers shown in Figure 5. All data transfers between the peripheral unit and the CPU use the data input and output registers. The handshake logic associated with each port controls the data transfers through the input and the output registers. The mode control register (two bits) selects one of the four programmable operating modes.

The control mode (Mode 3) uses the remaining registers. The input/output control register specifies which of the eight data bits in the port are to be outputs and enables these bits; the remaining bits are inputs. The mask register and the mask control register control Mode 3 interrupt conditions. The mask register specifies which of the bits in the port are active and which are masked or inactive.

The mask control register specifies two conditions: first, whether the active state of the input bits is High or Low, and second, whether an interrupt is generated when any one unmasked input bit is active (OR condition) or if the interrupt is generated when all unmasked input bits are active (AND condition).

Interrupt Control Logic. The interrupt control logic section handles all CPU interrupt protocol for nested-priority interrupt structures. Any device's physical location in a daisy-chain configuration determines its priority. Two lines (IEI and IEO) are provided in each PIO to form this daisy chain. The device closest to the CPU has the highest priority. Within a PIO, Port A interrupts have higher priority than those of Port B. In the byte input, byte output, or bidirectional modes, an interrupt can be generated whenever the peripheral requests a new byte transfer. In the bit control mode, an interrupt can be generated when the peripheral status matches a programmed value. The PIO provides for complete control of nested interrupts. That is, lower priority devices may not interrupt higher priority devices that have not had their interrupt service routines completed by the CPU. Higher priority devices may interrupt the servicing of lower priority devices.

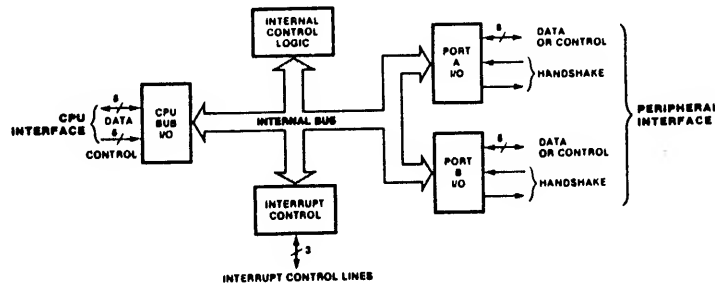


Figure 4. Block Diagram

Internal Structure
(Continued)

If the CPU (in interrupt Mode 2) accepts an interrupt, the interrupting device must provide an 8-bit interrupt vector for the CPU. This vector forms a pointer to a location in memory where the address of the interrupt service routine is located. The 8-bit vector from the interrupting device forms the least significant eight bits of the indirect pointer while the I Register in the CPU provides the most significant eight bits of the pointer. Each port (A and B) has an independent interrupt vector. The least significant bit of the vector is automatically set to 0 within the PIO because the pointer must point to two adjacent memory locations for a complete 16-bit address.

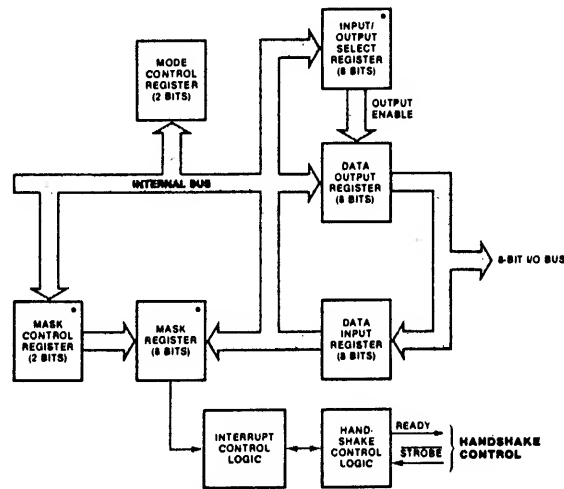
Unlike the other Z-80 peripherals, the PIO does not enable interrupts immediately after programming. It waits until $\overline{M}\overline{I}$ goes Low (e.g., during an opcode fetch). This condition is unimportant in the Z-80 environment but might not be if another type of CPU is used.

The PIO decodes the RETI (Return From

Interrupt) instruction directly from the CPU data bus so that each PIO in the system knows at all times whether it is being serviced by the CPU interrupt service routine. No other communication with the CPU is required.

CPU Bus I/O Logic. The CPU bus interface logic interfaces the Z-80 PIO directly to the Z-80 CPU, so no external logic is necessary. For large systems, however, address decoders and/or buffers may be necessary.

Internal Control Logic. This logic receives the control words for each port during programming and, in turn, controls the operating functions of the Z-80 PIO. The control logic synchronizes the port operations, controls the port mode, port addressing, selects the read/write function, and issues appropriate commands to the ports and the interrupt logic. The Z-80 PIO does not receive a write input from the CPU; instead, the \overline{RD} , \overline{CE} , $\overline{C/\overline{D}}$ and \overline{IORQ} signals generate the write input internally.



*Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state.

Figure 5. Typical Port I/O Block Diagram

Programming Mode 0, 1, or 2. (Byte Input, Output, or Bidirectional). Programming a port for Mode 0, 1, or 2 requires two words per port. These words are:

A Mode Control Word. Selects the port operating mode (Figure 6). This word may be written any time.

An Interrupt Vector. The Z-80 PIO is designed for use with the Z-80 CPU in interrupt Mode 2 (Figure 7). When interrupts are enabled, the PIO must provide an interrupt vector.

Mode 3. (Bit Input/Output). Programming a port for Mode 3 operation requires a control word, a vector (if interrupts are enabled), and three additional words, described as follows:

I/O Register Control. When Mode 3 is selected, the mode control word must be followed by another control word that sets the I/O control register, which in turn defines which port lines are inputs and which are outputs (Figure 8).

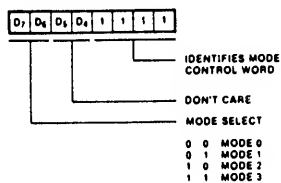


Figure 6. Mode Control Word

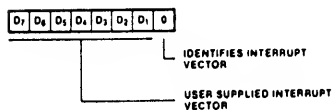


Figure 7. Interrupt Vector Word

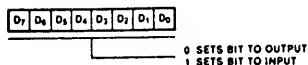
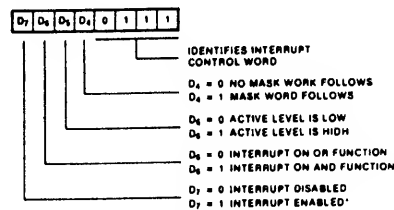


Figure 8. I/O Register Control Word

Interrupt Control Word. In Mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits changes to the active level, an interrupt is triggered). Bit D6 sets the logic function, as shown in Figure 9. The active level of the input bits can be set either High or Low. The active level is controlled by Bit D5.

Mask Control Word. This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, then D4 must be set. When D4 is set, the next word written to the port must be a mask control word (Figure 10).

Interrupt Disable. There is one other control word which can be used to enable or disable a port interrupt. It can be used without changing the rest of the interrupt control word (Figure 11).



*NOTE: THE PORT IS NOT ENABLED UNTIL THE INTERRUPT ENABLE IS FOLLOWED BY AN ACTIVE BIT

Figure 9. Interrupt Control Word

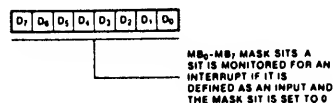


Figure 10. Mask Control Word

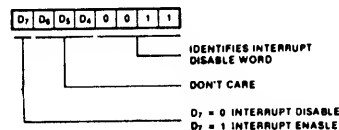


Figure 11. Interrupt Disable Word

A₀-A₇. Port A Bus (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port A of the PIO and a peripheral device. A₀ is the least significant bit of the Port A data bus.

ARDY. Register A Ready (output, active High). The meaning of this signal depends on the mode of operation selected for Port A as follows:

Output Mode. This signal goes active to indicate that the Port A output register has been loaded and the peripheral data bus is stable and ready for transfer to the peripheral device.

Input Mode. This signal is active when the Port A input register is empty and ready to accept data from the peripheral device.

Bidirectional Mode. This signal is active when data is available in the Port A output register for transfer to the peripheral device. In this mode, data is not placed on the Port A data bus, unless **ASTB** is active.

Control Mode. This signal is disabled and forced to a Low state.

ASTB. Port A Strobe Pulse From Peripheral Device (input, active Low). The meaning of this signal depends on the mode of operation selected for Port A as follows:

Output Mode. The positive edge of this strobe is issued by the peripheral to acknowledge the receipt of data made available by the PIO.

Input Mode. The strobe is issued by the peripheral to load data from the peripheral into the Port A input register. Data is loaded into the PIO when this signal is active.

Bidirectional Mode. When this signal is active, data from the Port A output register is gated onto the Port A bidirectional data bus. The positive edge of the strobe acknowledges the receipt of the data.

Control Mode. The strobe is inhibited internally.

B₀-B₇. Port B Bus (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port B and a peripheral device. The Port B data bus can supply 1.5 mA at 1.5 V to drive Darlington transistors. B₀ is the least significant bit of the bus.

B/ \bar{A} . Port B Or A Select (input, High = B). This pin defines which port is accessed during a data transfer between the CPU and the PIO. A Low on this pin selects Port A; a High selects Port B. Often address bit A₀ from the CPU is used for this selection function.

BRDY. Register B Ready (output, active High). This signal is similar to ARDY, except that in the Port A bidirectional mode this signal is High when the Port A input register is empty and ready to accept data from the peripheral device.

BSTB. Port B Strobe Pulse From Peripheral Device (input, active Low). This signal is similar to ASTB, except that in the Port A bidirectional mode this signal strobes data from the peripheral device into the Port A input register.

C/ \bar{D} . Control Or Data Select (input, High = C). This pin defines the type of data transfer to be performed between the CPU and the PIO. A High on this pin during a CPU write to the PIO causes the Z-80 data bus to be interpreted as a *command* for the port selected by the B/ \bar{A} Select line. A Low on this pin means that the Z-80 data bus is being used to transfer data between the CPU and the PIO. Often address bit A₁ from the CPU is used for this function.

$\bar{C}E$. Chip Enable (input, active Low). A Low on this pin enables the PIO to accept command or data inputs from the CPU during a write cycle or to transmit data to the CPU during a read cycle. This signal is generally decoded from four I/O port numbers for Ports A and B, data, and control.

CLK. System Clock (input). The Z-80 PIO uses the standard single-phase Z-80 system clock.

D₀-D₇. Z-80 CPU Data Bus (bidirectional, 3-state). This bus is used to transfer all data and commands between the Z-80 CPU and the Z-80 PIO. D₀ is the least significant bit.

IEI. Interrupt Enable In (input, active High). This signal is used to form a priority-interrupt daisy chain when more than one interrupt-driven device is being used. A High level on this pin indicates that no other devices of higher priority are being serviced by a CPU interrupt service routine.

IEO. Interrupt Enable Out (output, active High). The IEO signal is the other signal required to form a daisy chain priority scheme. It is High only if IEI is High and the CPU is not servicing an interrupt from this PIO. Thus this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

INT. Interrupt Request (output, open drain, active Low). When INT is active the Z-80 PIO is requesting an interrupt from the Z-80 CPU.

$\bar{I}ORQ$. Input/Output Request (input from Z-80 CPU, active Low). $\bar{I}ORQ$ is used in conjunction with B/ \bar{A} , C/ \bar{D} , $\bar{C}E$, and RD to transfer commands and data between the Z-80 CPU and the Z-80 PIO. When $\bar{C}E$, RD, and $\bar{I}ORQ$ are active, the port addressed by B/ \bar{A} transfers data to the CPU (a read operation). Conversely, when $\bar{C}E$ and $\bar{I}ORQ$ are active but RD is not, the port addressed by B/ \bar{A} is written into from the CPU with either data or control information, as specified by C/ \bar{D} . Also, if $\bar{I}ORQ$ and $\bar{M}\bar{I}$ are active simultaneously, the CPU is acknowledging an interrupt; the interrupting port automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

**Pin
Description
(Continued)**

$\overline{M1}$. *Machine Cycle* (input from CPU, active Low). This signal is used as a sync pulse to control several internal PIO operations. When both the $\overline{M1}$ and \overline{RD} signals are active, the Z-80 CPU is fetching an instruction from memory. Conversely, when both $\overline{M1}$ and \overline{IORQ} are active, the CPU is acknowledging an interrupt. In addition, $\overline{M1}$ has two other functions within the Z-80 PIO: it synchronizes

the PIO interrupt logic; when $\overline{M1}$ occurs without an active \overline{RD} or \overline{IORQ} signal, the PIO is reset.

\overline{RD} . *Read Cycle Status* (input from Z-80 CPU, active Low). If \overline{RD} is active, or an I/O operation is in progress, \overline{RD} is used with B/\overline{A} , C/\overline{D} , \overline{CE} , and \overline{IORQ} to transfer data from the Z-80 PIO to the Z-80 CPU.

Timing

The following timing diagrams show typical timing in a Z-80 CPU environment. For more precise specifications refer to the composite ac timing diagram.

Write Cycle. Figure 12 illustrates the timing for programming the Z-80 PIO or for writing data to one of its ports. No Wait states are allowed for writing to the PIO other than the automatically inserted T_{WA} . The PIO does not receive a specific write signal; it internally generates its own from the lack of an active \overline{RD} signal.

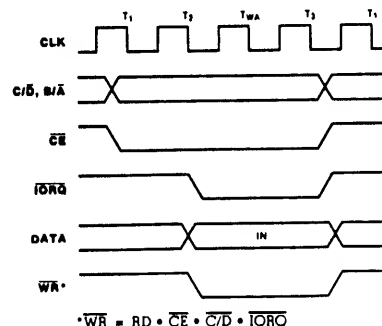


Figure 12. Write Cycle Timing

Read Cycle. Figure 13 illustrates the timing for reading the data input from an external device to one of the Z-80 PIO ports. No Wait states are allowed for reading the PIO other than the automatically inserted T_{WA} .

Output Mode (Mode 0). An output cycle (Figure 14) is always started by the execution of an output instruction by the CPU. The \overline{WR}^* pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The \overline{WR}^* pulse sets the Ready flag after a Low-going edge of CLK, indicating data is available. Ready stays active until the positive edge of the 'robe line is received, indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an \overline{INT} if the interrupt enable flip-flop has been set and if this device has the highest priority.

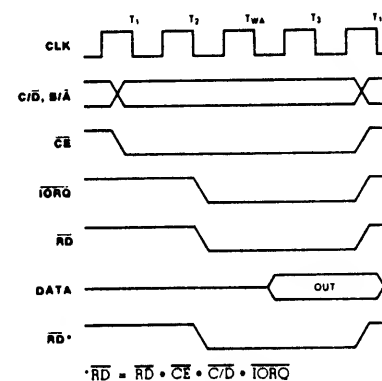


Figure 13. Read Cycle Timing

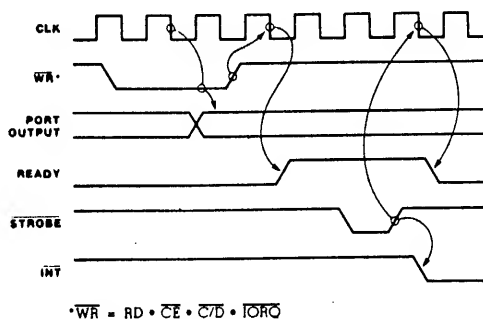


Figure 14. Mode 0 Output Timing

2006-0324, 0325, 0326

Timing
(Continued)

Input Mode (Mode 1). When $\overline{\text{STROBE}}$ goes Low, data is loaded into the selected port input register (Figure 15). The next rising edge of strobe activates INT , if Interrupt Enable is set and this is the highest-priority requesting device. The following falling edge of CLK resets Ready to an inactive state, indicating

that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete, the positive edge of RD sets Ready at the next Low-going transition of CLK . At this time new data can be loaded into the PIO.

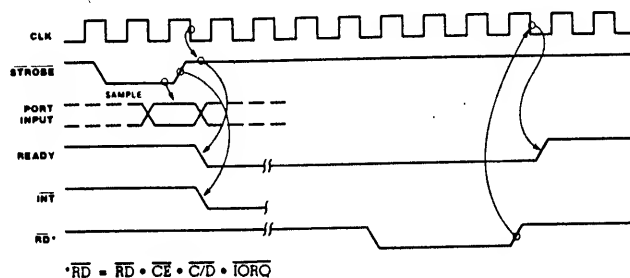


Figure 15. Mode 1 Input Timing

Bidirectional Mode (Mode 2). This is a combination of Modes 0 and 1 using all four handshake lines and the eight Port A I/O lines (Figure 16). Port B must be set to the bit mode and its inputs must be masked. The Port A handshake lines are used for output control and the Port B lines are used for input control.

If interrupts occur, Port A's vector will be used during port output and Port B's will be used during port input. Data is allowed out onto the Port A bus only when ASTB is Low. The rising edge of this strobe can be used to latch the data into the peripheral.

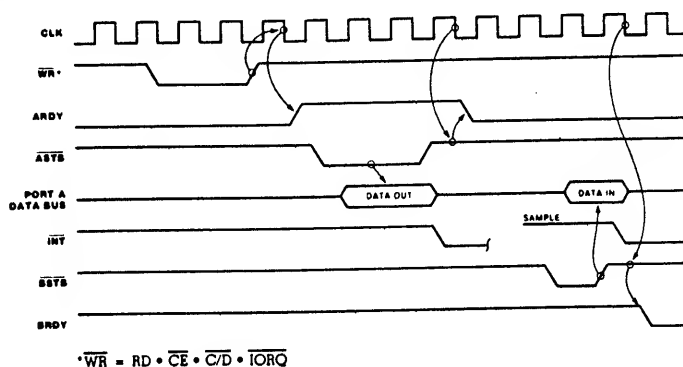


Figure 16. Mode 2 Bidirectional Timing

Timing (Continued)

Bit Mode (Mode 3). The bit mode does not utilize the handshake signals, and a normal port write or port read can be executed at any time. When writing, the data is latched into the output registers with the same timing as the output mode (Figure 17).

When reading the PIO, the data returned to the CPU is composed of output register data from those port data lines assigned as outputs and input register data from those port data

lines assigned as inputs. The input register contains data that was present immediately prior to the falling edge of \overline{RD} . An interrupt is generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers. However, if Port A is programmed in bidirectional mode, Port B does not issue an interrupt in bit mode and must therefore be polled.

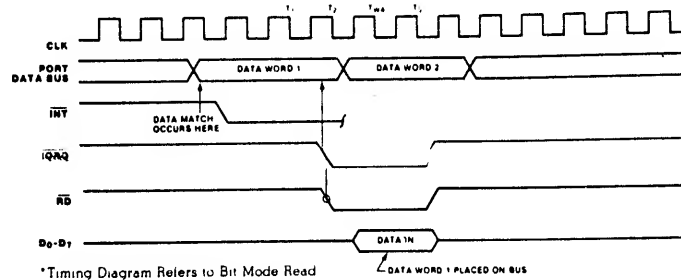


Figure 17. Mode 3 Bit Mode Timing

Interrupt Acknowledge Timing. During \overline{M} time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the Interrupt Enable signal to ripple through the daisy chain. The peripheral with IEI High and IEO Low during INTACK places a preprogrammed 8-bit interrupt vector on the data bus at this time (Figure 18). IEO is held Low until a Return From Interrupt (RETI) instruction is executed by the CPU while IEI is High. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.

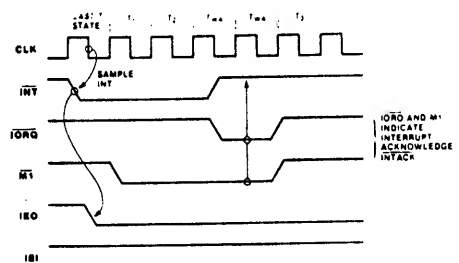


Figure 18. Interrupt Acknowledge Timing

Return From Interrupt Cycle. If a Z-80 peripheral has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always Low, inhibiting lower priority devices from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO is Low unless an "ED" is decoded as the first byte of a 2-byte opcode (Figure 19). In this case, IEO goes High until the next opcode byte is decoded, whereupon it goes Low again. If the second byte of the opcode was a "4D," then the opcode was an RETI instruction.

After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service has its IEI High and its

IEO Low. This device is the highest-priority device in the daisy chain that has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D," this peripheral device resets its "interrupt under service" condition.

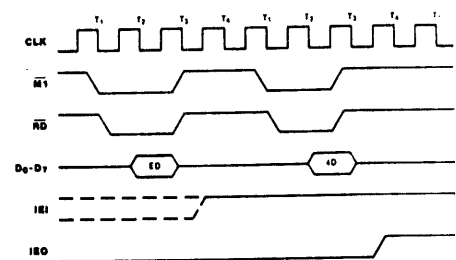
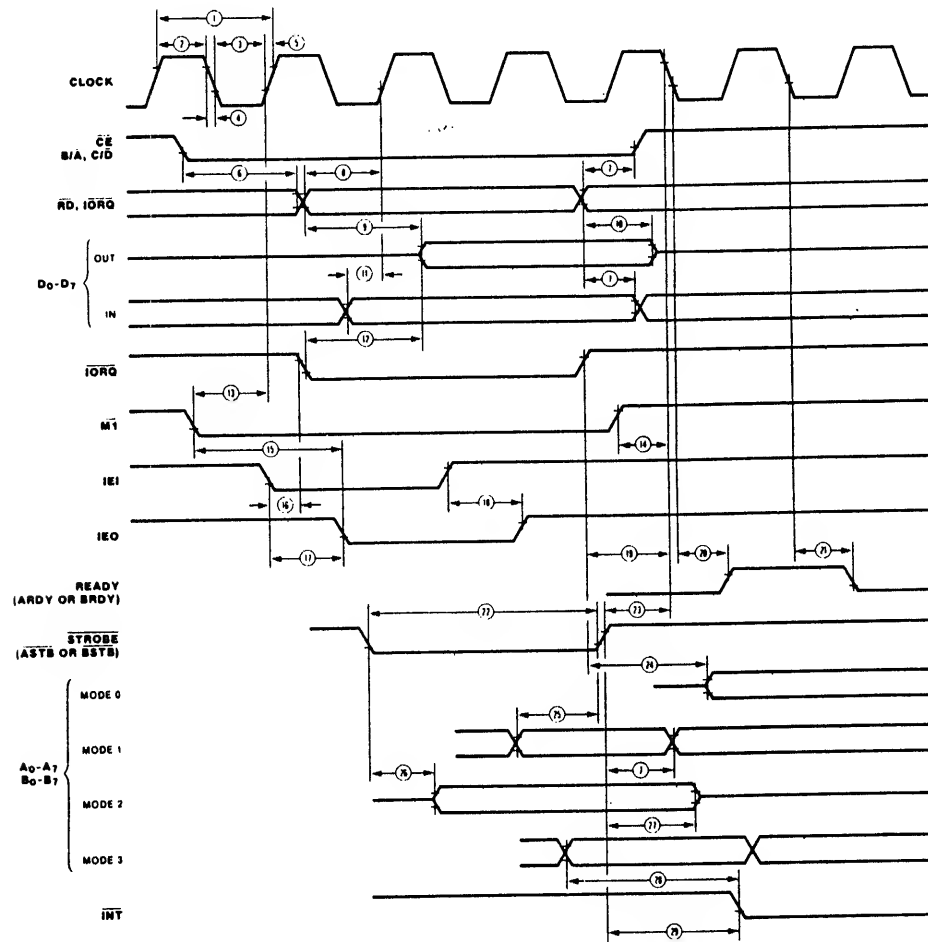


Figure 19. Return From Interrupt

AC
Charac-
teristics



2006-0332

ZILOG DATA
Z80 PIO

Number	Symbol	Parameter	Z-80 PIO		Z-80A PIO		Z-80B PIO ^[9]		Comment
			Min (ns)	Max (ns)	Min (ns)	Max (ns)	Min (ns)	Max (ns)	
1	T _c C	Clock Cycle Time	400	[1]	250	[1]	165	[1]	
2	TwCh	Clock Width (High)	170	2000	105	2000	65	2000	
3	TwCl	Clock Width (Low)	170	2000	105	2000	65	2000	
4	TIC	Clock Fall Time		30		30		20	
5	TrC	Clock Rise Time		30		30		20	
6	TsCS(RI)	\overline{CE} , B/A, C/D to \overline{RD} , \overline{IORQ} Setup Time	50		50		50		[6]
7	Th	Any Hold Times for Specified Setup Time	0		0		0	0	
8	TsRI(C)	\overline{RD} , \overline{IORQ} to Clock Setup Time	115		115		70		
9	TdRI(DO)	\overline{RD} , \overline{IORQ} to Data Out Delay		430		380		300	[2]
10	TdRI(DOs)	\overline{RD} , \overline{IORQ} to Data Out Float Delay		160		110		70	
11	TsDI(C)	Data In to Clock Setup Time	50		50		40		CL = 50 pF
12	TdIO(DOI)	\overline{IORQ} to Data Out Delay (INTACK Cycle)	340		160		120		[3]
13	TsMI(Cr)	\overline{MI} to Clock Setup Time	210		90		70		
14	TsMI(CI)	\overline{MI} to Clock Setup Time (MI Cycle)	0		0		0		[8]
15	TdMI(IEO)	\overline{MI} to IEO Delay (Interrupt Immediately Preceding MI)		300		190		100	[5, 7]
16	TsIEI(IO)	IEI to \overline{IORQ} Setup Time (INTACK Cycle)	140		140		100		[7]
17	TdIEI(IEOI)	IEI to IEO Delay		190		130		120	[5] CL = 50 pF
18	TdIEI(IEOIr)	IEI to IEO Delay (after ED Decode)		210		160		160	[5]
19	TcIO(C)	\overline{IORQ} to Clock Setup Time (To Activate READY on Next Clock Cycle)	220		200		170		
20	TdC(RDYr)	Clock to READY Delay	200		190		170		[5] CL = 50 pF
21	TdC(RDYI)	Clock to READY Delay	150		140		120		[5]
22	TwSTB	STROBE Pulse Width	150		150		120		[4]
23	TsSTB(C)	STROBE to Clock Setup Time (To Activate READY on Next Clock Cycle)	220		220		150		[5]
24	TdIO(PD)	\overline{IORQ} to PORT DATA Stable Delay (Mode 0)		200		180		160	[5]
25	TsPD(STB)	PORT DATA to STROBE Setup Time (Mode 1)	260		230		190		
26	TdSTB(PD)	STROBE to PORT DATA Stable (Mode 2)		230		210		180	[5]
27	TdSTB(PDr)	STROBE to PORT DATA Float Delay (Mode 2)		200		180		160	CL = 50 pF
28	TdPD(INT)	PORT DATA Match to \overline{INT} Delay (Mode 3)		540		490		430	
29	TdSTB(INT)	STROBE to \overline{INT} Delay		490		440		350	

NOTES

- [1] $T_cC = TwCh + TwCl + TrC + TIC$.
- [2] Increase TdRI(DO) by 10 ns for each 50 pF increase in load up to 200 pF max.
- [3] Increase TdIO(DOI) by 10 ns for each 50 pF increase in loading up to 200 pF max.
- [4] For Mode 2 $TwSTB > TsPD(STB)$.
- [5] Increase these values by 2 ns for each 10 pF increase in loading up to 100 pF max.

- [6] TsCS(RI) may be reduced. However, the time subtracted from TsCS(RI) will be added to TdRI(DO).
- [7] $2.5 T_cC > (N-2)TdIEI(IEOI) + TdMI(IEO) + TsIEI(IO) + TTL$ Buffer Delay, if any.
- [8] \overline{MI} must be active for a minimum of two clock cycles to reset the PIO.
- [9] Z80B PIO numbers are preliminary and subject to change.

ZILOG DATA Z80 PIO

Absolute Maximum Ratings

Voltages on all inputs and outputs
with respect to GND. -0.3 V to +7.0 V
Operating Ambient
Temperature As Specified in
Ordering Information
Storage Temperature -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Test Conditions

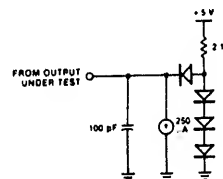
The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are:

- 0° to +70°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -40°C to +85°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -55° to +125°C,
+4.75 V ≤ V_{CC} ≤ +5.5 V

The product number for each operating temperature range may be found in the

Ordering Information section.

All ac parameters assume a load capacitance of 100 pF max. Timing references between two output signals assume a load difference of 50 pF max.



DC Charac- teristics

Symbol	Parameter	Min	Max	Unit	Test Condition
V _{IL}	Clock Input Low Voltage	-0.3	+0.45	V	
V _{IH}	Clock Input High Voltage	V _{CC} -0.6	+5.5	V	
V _{OL}	Input Low Voltage	0.3	+0.8	V	
V _{OH}	Input High Voltage	+2.0	+5.5	V	
V _{OL}	Output Low Voltage		+0.4	V	I _{OL} = 2.0 mA
V _{OH}	Output High Voltage	+2.4		V	I _{OH} = -250 μA
I _{IL}	Input Leakage Current	-10.0	+10.0	μA	0 < V _{IN} < V _{CC}
I _I	3 State Output/Data Bus Input Leakage Current	-10.0	+10.0	μA	0 < V _{IN} < V _{CC}
I _V	Power Supply Current		100.0	mA	V _{OH} = 1.5V
I _{OH}	Darlington Drive Current	-1.5	3.8	mA	R _{EXT} = 390 Ω

Over specified temperature range, 1 = 1MHz

Capacitance

Symbol	Parameter	Min	Max	Unit	Test Condition
C	Clock Capacitance		10	pF	Unmeasured pins returned to ground
C _{IN}	Input Capacitance		5	pF	
C _{OUT}	Output Capacitance		10	pF	

Over specified temperature range, 1 = 1MHz

08005 0006

ZILOG DATA
Z80 PIO

Z8430 Z80[®] CTC Counter/ Timer Circuit



Product Specification

March 1981

Features

- Four independently programmable counter/timer channels, each with a readable downcounter and a selectable 16 or 256 prescaler. Downcounters are reloaded automatically at zero count.
- Three channels have Zero Count/Timeout outputs capable of driving Darlington transistors.
- Selectable positive or negative trigger initiates timer operation.
- Standard Z-80 Family daisy-chain interrupt structure provides fully vectored, prioritized interrupts without external logic. The CTC may also be used as an interrupt controller.
- Interfaces directly to the Z-80 CPU or—for baud rate generation—to the Z-80 SIO.

General Description

The Z-80 CTC four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the Z-80 CTC satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

System design is simplified because the CTC connects directly to both the Z-80 CPU and the Z-80 SIO with no additional logic. In larger systems, address decoders and buffers may be required.

Programming the CTC is straightforward:

each channel is programmed with two bytes; a third is necessary when interrupts are enabled. Once started, the CTC counts down, reloads its time constant automatically, and resumes counting. Software timing loops are completely eliminated. Interrupt processing is simplified because only one vector need be specified; the CTC internally generates a unique vector for each channel.

The Z-80 CTC requires a single +5 V power supply and the standard Z-80 single-phase system clock. It is fabricated with n-channel silicon-gate depletion-load technology, and packaged in a 28-pin plastic or ceramic DIP.

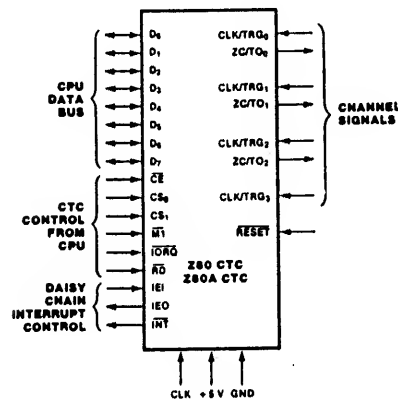


Figure 1. Pin Functions

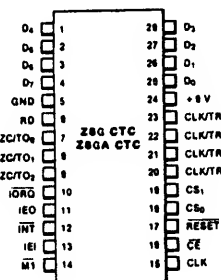


Figure 2. Pin Assignments

2041-0154, 0155

Functional Description

The Z-80 CTC has four independent counter/timer channels. Each channel is individually programmed with two words: a control word and a time-constant word. The control word selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. If the timing mode is selected, the control word also sets a prescaler, which divides the system clock by either 16 or 256. The time-constant word is a value from 1 to 256.

During operation, the individual counter channel counts down from the preset time constant value. In counter mode operation the counter decrements on each of the CLK/TRG input pulses until zero count is reached. Each decrement is synchronized by the system clock. For counts greater than 256, more than one counter can be cascaded. At zero count, the down-counter is automatically reset with the time constant value.

The timer mode determines time intervals as small as 4 μ s (Z-80A) or 6.4 μ s (Z-80) without additional logic or software timing loops. Time intervals are generated by dividing the system clock with a prescaler that decrements

a preset down-counter.

Thus, the time interval is an integral multiple of the clock period, the prescaler value (16 or 256) and the time constant that is preset in the down-counter. A timer is triggered automatically when its time constant value is programmed, or by an external CLK/TRG input.

Three channels have two outputs that occur at zero count. The first output is a zero-count/timeout pulse at the ZC/TO output. The fourth channel (Channel 3) does not have a ZC/TO output; interrupt request is the only output available from Channel 3.

The second output is Interrupt Request (INT), which occurs if the channel has its interrupt enabled during programming. When the Z-80 CPU acknowledges Interrupt Request, the Z-80 CTC places an interrupt vector on the data bus.

The four channels of the Z-80 CTC are fully prioritized and fit into four contiguous slots in a standard Z-80 daisy-chain interrupt structure. Channel 0 is the highest priority and Channel 3 the lowest. Interrupts can be individually enabled (or disabled) for each of the four channels.

Architecture

The CTC has four major elements, as shown in Figure 3.

- CPU bus I/O
- Channel control logic
- Interrupt logic
- Counter/timer circuits

CPU Bus I/O. The CPU bus I/O circuit decodes the address inputs, and interfaces the CPU data and control signals to the CTC for distribution on the internal bus.

Internal Control Logic. The CTC internal control logic controls overall chip operating functions such as the chip enable, reset, and read/write logic.

Interrupt Logic. The interrupt control logic ensures that the CTC interrupts interface properly with the Z-80 CPU interrupt system. The logic controls the interrupt priority of the CTC as a function of the IEI signal. If IEI is High, the CTC has priority. During interrupt

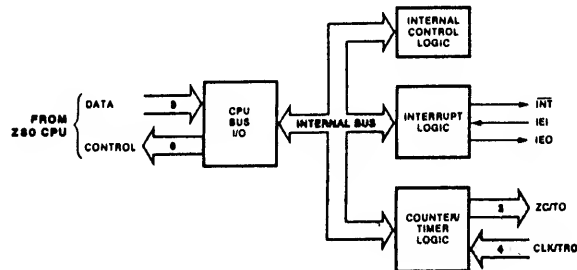


Figure 3. Functional Block Diagram

2041-0157

Architecture (Continued)

processing, the interrupt logic holds IEO Low, which inhibits the interrupt operation on lower priority devices. If the IEI input goes Low, priority is relinquished and the interrupt logic drives IEO Low.

If a channel is programmed to request an interrupt, the interrupt logic drives IEO Low at the zero count, and generates an INT signal to the Z-80 CPU. When the Z-80 CPU responds with interrupt acknowledge (MI and IORQ), then the interrupt logic arbitrates the CTC internal priorities, and the interrupt control logic places a unique interrupt vector on the data bus.

If an interrupt is pending, the interrupt logic holds IEO Low. When the Z-80 CPU issues a Return From Interrupt (RETI) instruction, each peripheral device decodes the first byte (ED₁₆). If the device has a pending interrupt, it raises IEO (High) for one M1 cycle. This ensures that all lower priority devices can decode the entire RETI instruction and reset properly.

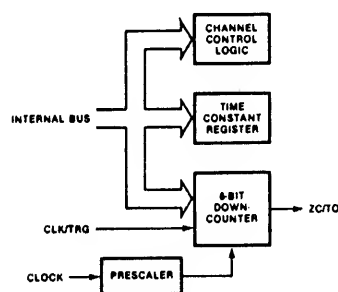


Figure 4. Counter/Timer Block Diagram

Counter/Timer Circuits. The CTC has four independent counter/timer circuits, each containing the logic shown in Figure 4.

Channel Control Logic. The channel control logic receives the 8-bit channel control word when the counter/timer channel is programmed. The channel control logic decodes

the control word and sets the following operating conditions:

- Interrupt enable (or disable)
- Operating mode (timer or counter)
- Timer mode prescaler factor (16 or 256)
- Active slope for CLK/TRG input
- Timer mode trigger (automatic or CLK/TRG input)
- Time constant data word to follow
- Software reset

Time Constant Register. When the counter/timer channel is programmed, the time constant register receives and stores an 8-bit time constant value, which can be anywhere from 1 to 256 ($0 = 256$). This constant is automatically loaded into the down-counter when the counter/timer channel is initialized, and subsequently after each zero count.

Prescaler. The prescaler, which is used only in timer mode, divides the system clock frequency by a factor of either 16 or 256. The prescaler output clocks the down-counter during timer operation. The effect of the prescaler on the down-counter is a multiplication of the system clock period by 16 or 256. The prescaler factor is programmed by bit 5 of the channel control word.

Down-Counter. Prior to each count cycle, the down-counter is loaded with the time constant register contents. The counter is then decremented one of two ways, depending on operating mode:

- By the prescaler output (timer mode)
- By the trigger pulses into the CLK/TRG input (counter mode)

Without disturbing the down-count, the Z-80 CPU can read the count remaining at any time by performing an I/O read operation at the port address assigned to the CTC channel. When the down-counter reaches the zero count, the ZC/TO output generates a positive-going pulse. When the interrupt is enabled, zero count also triggers an interrupt request signal (INT) from the interrupt logic.

Programming Each Z-80 CTC channel must be programmed prior to operation. Programming consists of writing two words to the I/O port that corresponds to the desired channel. The first word is a control word that selects the operating mode and other parameters; the second word is a time constant, which is a binary data word with a value from 1 to 256. A time constant word must be preceded by a channel control word.

After initialization, channels may be reprogrammed at any time. If updated control and time constant words are written to a channel during the count operation, the count continues to zero before the new time constant is loaded into the counter.

If the interrupt on any Z-80 CTC channel is enabled, the programming procedure should also include an interrupt vector. Only one vector is required for all four channels, because the interrupt logic automatically modifies the vector for the channel requesting service.

A control word is identified by a 1 in bit 0. A 0 in bit 2 indicates a time constant word is to follow. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

Addressing. During programming, channels are addressed with the channel select pins CS₁ and CS₀. A 2-bit binary code selects the appropriate channel as shown in the following table.

Channel	CS ₁	CS ₀
0	0	0
1	0	1
2	1	0
3	1	1

Reset. The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZC/TO and Interrupt outputs go inactive, IEO reflects IEI, and

D₀-D₇ go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

The software reset is controlled by bit 1 in the channel control word. When a channel receives a software reset, it stops counting. When a software reset is used, the other bits in the control word also change the contents of the channel control register. After a software reset a new time constant word must be written to the same channel.

If the channel control word has both bits D₁ and D₂ set to 1, the addressed channel stops operating, pending a new time constant word. The channel is ready to resume after the new constant is programmed. In timer mode, if D₃ = 0, operation is triggered automatically when the time constant word is loaded.

Channel Control Word Programming. The channel control word is shown in Figure 5. It sets the modes and parameters described below.

Interrupt Enable. D₇ enables the interrupt, so that an interrupt output (INT) is generated at zero count. Interrupts may be programmed in either mode and may be enabled or disabled at any time.

Operating Mode. D₆ selects either timer or counter mode.

Prescaler Factor. (Timer Mode Only). D₅ selects factor—either 16 or 256.

Trigger Slope. D₄ selects the active edge or slope of the CLK/TRG input pulses. Note that reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as a CLK/TRG pulse and the timer starts. Similarly, if the channel is in counter mode, the counter decrements.

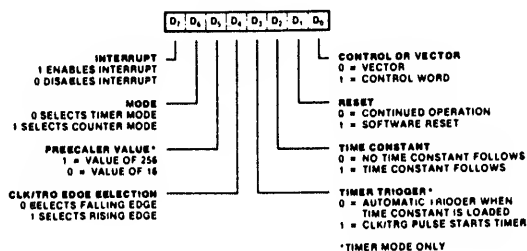


Figure 5. Channel Control Word

Programming
(Continued)

Trigger Mode (Timer Mode Only). D_3 selects the trigger mode for timer operation. When D_3 is reset to 0, the timer is triggered automatically. The time constant word is programmed during an I/O write operation, which takes one machine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse (T_2) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer reloads automatically and continues counting without interruption or delay, until stopped by a reset.

When D_3 is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, which takes one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse (T_2) of the following machine cycle. Note that the first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede T_2 by one clock cycle plus its minimum setup time. If the minimum time is not met, the timer will start on the third clock cycle (T_3).

Once started the timer operates continuously, without interruption or delay, until stopped by a reset.

Time Constant to Follow. A 1 in D_2 indicates that the next word addressed to the selected channel is a time constant data word for the time constant register. The time constant word may be written at any time.

A 0 in D_2 indicates no time constant word is to follow. This is ordinarily used when the channel is already in operation and the new channel control word is an update. A channel will not operate without a time constant value. The only way to write a time constant value is to write a control word with D_2 set.

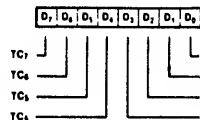


Figure 6. Time Constant Word

Software Reset. Setting D_1 to 1 causes a software reset, which is described in the Reset section.

Control Word. Setting D_0 to 1 identifies the word as a control word.

Time Constant Programming. Before a channel can start counting it must receive a time constant word from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant word to indicate that the next word is a time constant. The time constant word can be any value from 1 to 256 (Figure 6). Note that 00_{16} is interpreted as 256.

In timer mode, the time interval is controlled by three factors:

- The system clock period (ϕ)
- The prescaler factor (P), which multiplies the interval by either 16 or 256
- The time constant (T), which is programmed into the time constant register

Consequently, the time interval is the product of $\phi \times P \times T$. The minimum timer resolution is $16 \times \phi$ (4 μ s with a 4 MHz clock). The maximum timer interval is $256 \times \phi \times 256$ (16.4 ms with a 4 MHz clock). For longer intervals timers may be cascaded.

Interrupt Vector Programming. If the Z-80 CTC has one or more interrupts enabled, it can supply interrupt vectors to the Z-80 CPU. To do so, the Z-80 CTC must be pre-programmed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the Z-80 CTC Channel 0. Note that D_0 of the vector word is always zero, to distinguish the vector from a channel control word. D_1 and D_2 are not used in programming the vector word. These bits are supplied by the interrupt logic to identify the channel requesting interrupt service with a unique interrupt vector (Figure 7). Channel 0 has the highest priority.

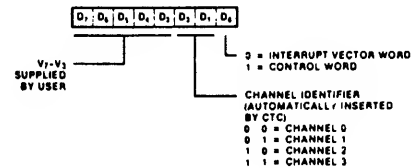


Figure 7. Interrupt Vector Word

Timing

Read Cycle Timing. Figure 9 shows read cycle timing. This cycle reads the contents of a down-counter without disturbing the count. During clock cycle T_2 , the Z-80 CPU initiates a read cycle by driving the following inputs Low: \overline{RD} , \overline{IORQ} , and \overline{CE} . A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be read. $M1$ must be High to distinguish this cycle from an interrupt acknowledge. No additional wait states are allowed.

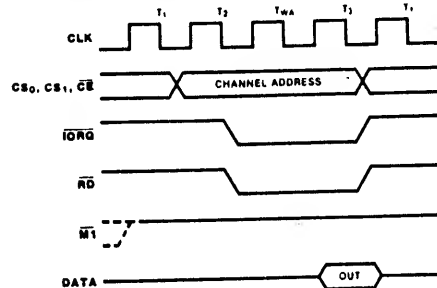


Figure 9. Read Cycle Timing

Write Cycle Timing. Figure 10 shows write cycle timing for loading control, time constant or vector words.

The CTC does not have a write signal input, so it generates one internally when the read (\overline{RD}) input is High during T_1 . During T_2 \overline{IORQ} and \overline{CE} inputs are Low. $M1$ must be High to distinguish a write cycle from an interrupt acknowledge. A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be addressed, and the word being written is placed on the Z-80 data bus. The data word is

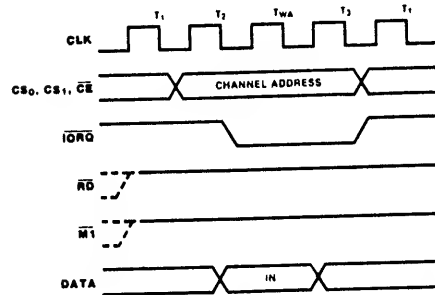


Figure 10. Write Cycle Timing

latched into the appropriate register with the rising edge of clock cycle T_{WA} . No additional wait states are allowed.

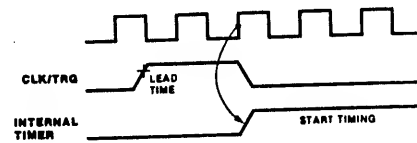


Figure 11. Timer Mode Timing

Timer Operation. In the timer mode, a CLK/TRG pulse input starts the timer (Figure 11) on the second succeeding rising edge of CLK. The trigger pulse is asynchronous and it must have a minimum width. A minimum lead time (210 ns) is required between the active edge of the CLK/TRG and the next rising edge of CLK to enable the prescaler on the following clock edge. If the CLK/TRG edge occurs closer than this, the initiation of the timer function is delayed one clock cycle. This corresponds to the startup timing discussed in the programming section. The timer can also be started automatically if so programmed by the channel control word.

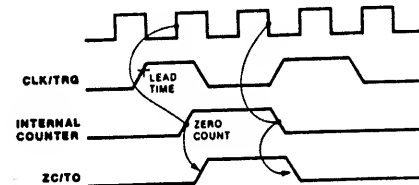


Figure 12. Counter Mode Timing

Counter Operation. In the counter mode, the CLK/TRG pulse input decrements the down-counter. The trigger is asynchronous, but the count is synchronized with CLK. For the decrement to occur on the next rising edge of CLK, the trigger edge must precede CLK by a minimum lead time as shown in Figure 12. If the lead time is less than specified, the count is delayed by one clock cycle. The trigger pulse must have a minimum width, and the trigger period must be at least twice the clock period.

The ZC/TO output occurs immediately after zero count, and follows the rising CLK edge.

Interrupt Operation

The Z-80 CTC follows the Z-80 system interrupt protocol for nested priority interrupts and return from interrupt, wherein the interrupt priority of a peripheral is determined by its location in a daisy chain. Two lines—IEI and IEO—in the CTC connect it to the system daisy chain. The device closest to the +5 V supply has the highest priority (Figure 13). For additional information on the Z-80 interrupt structure, refer to the *Z-80 CPU Product Specification* and the *Z-80 CPU Technical Manual*.

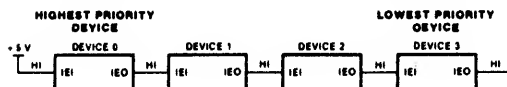


Figure 13. Daisy-Chain Interrupt Priorities

Within the Z-80 CTC, interrupt priority is predetermined by channel number: Channel 0 has the highest priority, and Channel 3 the lowest. If a device or channel is being serviced with an interrupt routine, it cannot be interrupted by a device or channel with lower priority until service is complete. Higher priority devices or channels may interrupt the servicing of lower priority devices or channels.

A Z-80 CTC channel may be programmed to request an interrupt every time its down-counter reaches zero. Note that the CPU must be programmed for interrupt mode 2. Some time after the interrupt request, the CPU sends an interrupt acknowledge. The CTC interrupt control logic determines the highest priority channel that is requesting an interrupt. Then, if the CTC IEI input is High (indicating that it has priority within the system daisy chain) it places an 8-bit interrupt vector on the system data bus. The high-order five bits of this vector

were written to the CTC during the programming process; the next two bits are provided by the CTC interrupt control logic as a binary code that identifies the highest priority channel requesting an interrupt; the low-order bit is always zero.

Interrupt Acknowledge Timing. Figure 14 shows interrupt acknowledge timing. After an interrupt request, the Z-80 CPU sends an interrupt acknowledge (\overline{MI} and \overline{IORQ}). All channels are inhibited from changing their interrupt request status when \overline{MI} is active—about two clock cycles earlier than \overline{IORQ} . \overline{RD} is High to distinguish this cycle from an instruction fetch.

The CTC interrupt logic determines the highest priority channel requesting an interrupt. If the CTC interrupt enable input (IEI) is High, the highest priority interrupting channel within the CTC places its interrupt vector on the data bus when \overline{IORQ} goes Low. Two wait states (T_{WA}) are automatically inserted at this time to allow the daisy chain to stabilize. Additional wait states may be added.

Return from Interrupt Timing. At the end of an interrupt service routine the RETI (Return From Interrupt) instruction initializes the daisy chain enable lines for proper control of nested priority interrupt handling. The CTC decodes the 2-byte RETI code internally and determines whether it is intended for a channel being serviced. Figure 15 shows RETI timing.

If several Z-80 peripherals are in the daisy chain, IEI settles active (High) on the chip currently being serviced when the opcode ED_{16} is decoded. If the following opcode is $4D_{16}$, the peripheral being serviced is released and its IEO becomes active. Additional wait states are allowed.

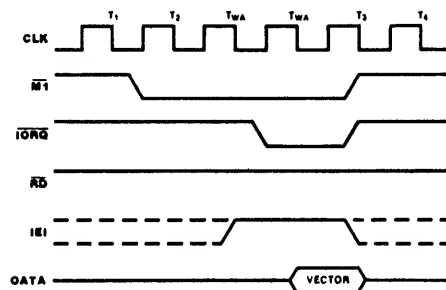


Figure 14. Interrupt Acknowledge Timing

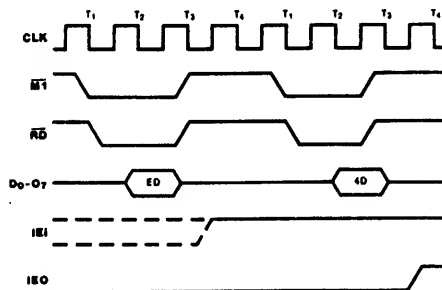


Figure 15. Return From Interrupt Timing

Absolute Maximum Ratings

Voltages on all inputs and outputs with respect to GND.....-0.3 V to +7.0 V

Operating Ambient TemperatureAs Specified in Ordering Information

Storage Temperature-65°C to +150°C

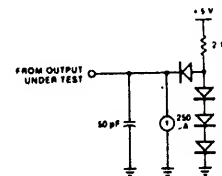
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Test Conditions

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are:

- 0°C to +70°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -40°C to +85°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -55°C to +125°C,
+4.5 V ≤ V_{CC} ≤ +5.5 V

The product number for each operating temperature range may be found in the ordering information section.



DC Characteristics	Symbol	Parameter	Min	Max	Unit	Test Condition
	V _{ILC}	Clock Input Low Voltage	-0.3	+0.45	V	
	V _{IHC}	Clock Input High Voltage	V _{CC} - .6	V _{CC} + .3	V	
	V _{IL}	Input Low Voltage	-0.3	+0.8	V	
	V _{IH}	Input High Voltage	+2.0	V _{CC}	V	
	V _{OL}	Output Low Voltage		+0.4	V	I _{OL} = 2 mA
	V _{OH}	Output High Voltage	+2.4		V	I _{OH} = 250 μA
	I _{CC}	Power Supply Current		+120	mA	
	I _{LI}	Input Leakage Current		+10	μA	V _{IN} = 0 to V _{CC}
	I _{LOH}	3-State Output Leakage Current in Float		+10	μA	V _{OUT} = 2.4 to V _{CC}
	I _{LOL}	3-State Output Leakage Current in Float		-10	μA	V _{OUT} = 0.4 V
	I _{OHD}	Darlington Drive Current	-1.5		mA	V _{OH} = 1.5 V R _{EXT} = 390Ω

Capacitance	Symbol	Parameter	Max	Unit	Condition
	CLK	Clock Capacitance	20	pF	Unmeasured pins returned to ground
	C _{IN}	Input Capacitance	5	pF	
	C _{OUT}	Output Capacitance	10	pF	

T_A = 25°C, f = 1 MHz

The timing diagram illustrates the relationship between the 8086 microprocessor and its external components. The signals shown are:

- CLOCK**: The system clock signal.
- CS₀, CS₁**: Chip select signals for memory banks.
- \overline{CE}** : Active-low chip enable signal.
- \overline{IORQ}** : Active-low input/output request signal.
- \overline{RD}** : Active-low read signal.
- DATA**: The data bus signal.
- CS₀, CS₁**: Chip select signals for memory banks (repeated).
- \overline{CE}** : Active-low chip enable signal (repeated).
- \overline{IORQ}** : Active-low input/output request signal (repeated).
- DATA**: The data bus signal (repeated).
- $\overline{M1}$** : Active-low master/slave select signal.
- \overline{IORQ}** : Active-low input/output request signal (repeated).
- DATA**: The data bus signal (repeated).
- \overline{IEI}** : Active-low interrupt enable signal.
- \overline{IEO}** : Active-low interrupt enable signal.
- \overline{INT}** : Active-low interrupt signal.
- K/TRAQ₀₋₃ (COUNTER MODE)**: Counter mode signals.
- K/TRAQ₀₋₃ (TIMER MODE)**: Timer mode signals.
- ZC/TAQ₀₋₃**: Zero compare/timer mode signals.

Timing parameters are indicated by arrows and labeled with letters:

- 1**: Clock period.
- 2**: Address setup time before CS or \overline{CE} .
- 3**: Address hold time after CS or \overline{CE} .
- 4**: Data setup time before \overline{IORQ} .
- 5**: Data hold time after \overline{IORQ} .
- 6**: \overline{RD} setup time before \overline{IORQ} .
- 7**: \overline{RD} hold time after \overline{IORQ} .
- 8**: \overline{RD} setup time before $\overline{M1}$.
- 9**: \overline{RD} hold time after $\overline{M1}$.
- 10**: \overline{RD} setup time before \overline{IEI} .
- 11**: \overline{RD} hold time after \overline{IEI} .
- 12**: \overline{RD} setup time before \overline{IEO} .
- 13**: \overline{RD} hold time after \overline{IEO} .
- 14**: \overline{RD} setup time before \overline{INT} .
- 15**: \overline{RD} hold time after \overline{INT} .
- 16**: \overline{RD} setup time before K/TRAQ₀₋₃ (COUNTER MODE).
- 17**: \overline{RD} hold time after K/TRAQ₀₋₃ (COUNTER MODE).
- 18**: \overline{RD} setup time before K/TRAQ₀₋₃ (TIMER MODE).
- 19**: \overline{RD} hold time after K/TRAQ₀₋₃ (TIMER MODE).
- 20**: \overline{RD} setup time before ZC/TAQ₀₋₃.
- 21**: \overline{RD} hold time after ZC/TAQ₀₋₃.

5 - 46

Number	Symbol	Parameter	Z-80 CTC		Z-80A CTC		Z-80B CTC		Notes
			Min (ns)	Max (ns)	Min (ns)	Max (ns)	Min (ns)	Max (ns)	
1	T _c C	Clock Cycle Time	400	[1]	250	[1]	165	[1]	
2	T _w CH	Clock Width (High)	170	2000	105	2000	65	2000	
3	T _w Cl	Clock Width (Low)	170	2000	105	2000	65	2000	
4	T _f C	Clock Fall Time		30		30		20	
5	T _r C	Clock Rise Time		30		30		20	
6	T _h	All Hold Times	0		0		0		
7	T _s CS(C)	CS to Clock ↑ Setup Time	250		160		100		
8	T _s CE(C)	\overline{CE} to Clock ↑ Setup Time	200		150		100		
9	T _s IO(C)	\overline{IORQ} ↓ to Clock ↑ Setup Time	250		115		70		
10	T _s RD(C)	\overline{RD} ↓ to Clock ↑ Setup Time	240		115		70		
11	T _d C(DO)	Clock ↑ to Data Out Delay		240		200		130	[2]
12	T _d C(DOz)	Clock ↑ to Data Out Float Delay		230		110		90	
13	T _s DI(C)	Data In to Clock ↑ Setup Time	60		50		40		
14	T _s M1(C)	$\overline{M1}$ ↓ to Clock ↑ Setup Time	210		90		70		
15	T _d M1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (Interrupt immediately preceding $\overline{M1}$)		300		190		130	[3]
16	T _d IO(DOI)	\overline{IORQ} ↓ to Data Out Delay (INTA Cycle)		340		160		110	[2]
17	T _d IEI(IEOI)	IEI ↓ to IEO ↓ Delay		190		130		100	[3]
18	T _d IEI(IEOr)	IEI ↓ to IEO ↓ Delay (After ED Decode)		220		160		110	[3]
19	T _d C(INT)	Clock ↑ to \overline{INT} ↓ Delay		(T _c C + 200)		(T _c C + 140)		T _c C + 120	[4]
20	T _d CLK(INT)	CLK/TRG ↑ to \overline{INT} ↓ tsCTR(C) satisfied tsCTR(C) not satisfied		(T _c C + 230) (2T _c C + 530)		(T _c C + 160) (2T _c C + 370)		T _c C + 130 2T _c C + 280	[5] [5] [5]
21	T _c CTR	CLK/TRG Cycle Time		(2T _c C)		(2T _c C)		2T _c C	
22	T _r CTR	CLK/TRG Rise Time		50		50		40	
23	T _f CTR	CLK/TRG Fall Time		50		50		40	
24	T _w CTRl	CLK/TRG Width (Low)	200		200		120		
25	T _w CTRh	CLK/TRG Width (High)	200		200		120		
26	T _s CTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for Immediate Count	300		210		150		[5]
27	T _s CTR(Ct)	CLK/TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following clock ↑	210		210		150		[4]
28	T _d C(ZC/TOr)	Clock ↑ to ZC/TO ↓ Delay		260		190		140	
29	T _d C(ZC/TOl)	Clock ↑ to ZC/TO ↓ Delay		190		190		140	

[A]: $2.5 T_{cC} > (n-2) T_{dIEI(IEOI)} + T_{dM1(IEO)} + T_{sIEI(IEO)}$
+ TTL buffer delay, if any.

[B]: RESET must be active for a minimum of 3 clock cycles.

NOTES:

[1]: $T_{cC} = T_{wCH} + T_{wCl} + T_{rC} + T_{fC}$.

[2]: Increase delay by 10 ns for each 50 pF increase in loading.
200 pF maximum for data lines, and 100 pF for control lines.

[3]: Increase delay by 2 ns for each 10 pF increase in loading.
100 pF maximum.

[4]: Timer mode.

[5]: Counter mode.

[6]: RESET must be active for a minimum of 3 clock cycles.

ZILOG DATA
Z80 CTC

Ordering Information	Product Number	Package/Temp	Speed	Description	Product Number	Package/Temp	Speed	Description
	Z8430	CE	2.5 MHz	Z80 CTC (28-pin)	Z8430A	DE	4.0 MHz	Z80A CTC (28-pin)
	Z8430	CM	2.5 MHz	Same as above	Z8430A	DS	4.0 MHz	Same as above
	Z8430	CMB	2.5 MHz	Same as above	Z8430A	PE	4.0 MHz	Same as above
	Z8430	CS	2.5 MHz	Same as above	Z8430A	PS	4.0 MHz	Same as above
	Z8430	DE	2.5 MHz	Same as above	Z8430B	CE	6.0 MHz	Z80B CTC (28-pin)
	Z8430	DS	2.5 MHz	Same as above	Z8430B	CM	6.0 MHz	Same as above
	Z8430	PE	2.5 MHz	Same as above	Z8430B	CMB	6.0 MHz	Same as above
	Z8430	PS	2.5 MHz	Same as above	Z8430B	CS	6.0 MHz	Same as above
	Z8430A	CE	4.0 MHz	Z80A CTC (28-pin)	Z8430B	DE	6.0 MHz	Same as above
	Z8430A	CM	4.0 MHz	Same as above	Z8430B	DS	6.0 MHz	Same as above
	Z8430A	CMB	4.0 MHz	Same as above	Z8430B	PE	6.0 MHz	Same as above
	Z8430A	CS	4.0 MHz	Same as above	Z8430B	PS	6.0 MHz	Same as above

NOTES: C = Ceramic, D = Cerdip, P = Plastic; E = -40°C to +85°C; M = -55°C to +125°C; MB = -55°C to +125°C with MIL-STD-883 Class B processing; S = 0°C to +70°C.

00-2022-A

ZILOG DATA
Z80 CTC

Z8440 Z80[®] SIO Serial Input/Output Controller



Product Specification

March 1981

Features

- Two independent full-duplex channels, with separate control and status lines for modems or other devices.
- Data rates of 0 to 500K bits/second in the x1 clock mode with a 2.5 MHz clock (Z-80 SIO), or 0 to 800K bits/second with a 4.0 MHz clock (Z-80A SIO).
- Asynchronous protocols: everything necessary for complete messages in 5, 6, 7 or 8 bits/character. Includes variable stop bits and several clock-rate multipliers; break generation and detection; parity; overrun and framing error detection.
- Synchronous protocols: everything necessary for complete bit- or byte-oriented messages in 5, 6, 7 or 8 bits/character, including IBM Bisync, SDLC, HDLC, CCITT-X.25 and others. Automatic CRC generation/checking, sync character and zero insertion/deletion, abort generation/detection and flag insertion.
- Receiver data registers quadruply buffered, transmitter registers doubly buffered.
- Highly sophisticated and flexible daisy-chain interrupt vectoring for interrupts without external logic.

General Description

The Z-80 SIO Serial Input/Output Controller is a dual-channel data communication interface with extraordinary versatility and capability. Its basic functions as a serial-to-parallel, parallel-to-serial converter/controller can be programmed by a CPU for a broad range of serial communication applications. The device supports all common asynchronous and synchronous protocols, byte- or

bit-oriented, and performs all of the functions traditionally done by UARTs, USARTs and synchronous communication controllers combined, plus additional functions traditionally performed by the CPU. Moreover, it does this on two fully-independent channels, with an exceptionally sophisticated interrupt structure that allows very fast transfers. Full interfacing is provided for CPU or DMA

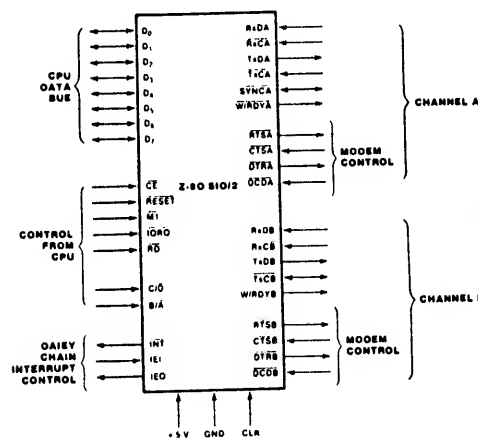


Figure 1. Z-80 SIO/2 Pin Functions

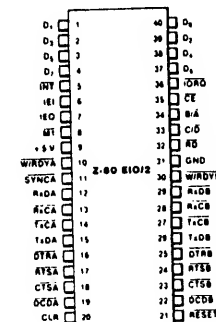


Figure 2. Z-80 SIO/2 Pin Assignments

2042 0111 0120

General Description
(Continued)

control. In addition to data communication, the circuit can handle virtually all types of serial I/O with fast (or slow) peripheral devices. While designed primarily as a member of the Z-80 family, its versatility makes it well suited to many other CPUs.

The Z-80 SIO is an n-channel silicon-gate depletion-load device packaged in a 40-pin plastic or ceramic DIP. It uses a single +5 V power supply and the standard Z-80 family single-phase clock.

Pin Description

Figures 1 through 6 illustrate the three pin configurations (bonding options) available in the SIO. The constraints of a 40-pin package make it impossible to bring out the Receive Clock (RxC), Transmit Clock (TxCl), Data Terminal Ready (DTR) and Sync (SYNC) signals for both channels. Therefore, either Channel B lacks a signal or two signals are bonded together in the three bonding options offered:

- Z-80 SIO/2 lacks SYNCB
- Z-80 SIO/1 lacks DTRB
- Z-80 SIO/0 has all four signals, but TxClB and RxClB are bonded together

The first bonding option above (SIO/2) is the preferred version for most applications. The pin descriptions are as follows:

B/A. Channel A Or B Select (input, High selects Channel B). This input defines which channel is accessed during a data transfer between the CPU and the SIO. Address bit A₀ from the CPU is often used for the selection function.

C/D. Control Or Data Select (input, High selects Control). This input defines the type of information transfer performed between the CPU and the SIO. A High at this input during a CPU write to the SIO causes the information on the data bus to be interpreted as a command for the channel selected by B/A. A Low at C/D means that the information on the data bus is data. Address bit A₁ is often used for this function.

CE. Chip Enable (input, active Low). A Low level at this input enables the SIO to accept command or data input from the CPU during a write cycle or to transmit data to the CPU during a read cycle.

CLK. System Clock (input). The SIO uses the standard Z-80 System Clock to synchronize internal signals. This is a single-phase clock.

CTSA, CTSB. Clear To Send (inputs, active Low). When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these inputs and interrupts the CPU on both logic level transitions. The Schmitt-trigger buffering does not guarantee a specified noise-level margin.

D₀-D₇. System Data Bus (bidirectional, 3-state). The system data bus transfers data and commands between the CPU and the Z-80 SIO. D₀ is the least significant bit.

DCDA, DCDB. Data Carrier Detect (inputs, active Low). These pins function as receiver enables if the SIO is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these pins and interrupts the CPU on both logic level transitions. Schmitt-trigger buffer-

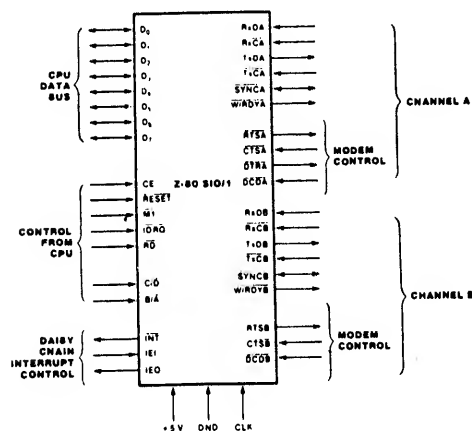


Figure 3. Z-80 SIO/1 Pin Functions

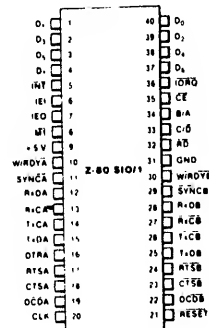


Figure 4. Z-80 SIO/1 Pin Assignments

2042 0111 0120

**Pin
Description**
(Continued)

ing does not guarantee a specific noise-level margin.

DTRA, DTRB. *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into Z-80 SIO. They can also be programmed as general-purpose outputs.

In the Z-80 SIO/I bonding option, **DTRB** is omitted.

IEI. *Interrupt Enable In* (input, active High). This signal is used with **IEO** to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (output, active High). **IEO** is High only if **IEI** is High and the CPU is not servicing an interrupt from this SIO. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

INT. *Interrupt Request* (output, open drain, active Low). When the SIO is requesting an interrupt, it pulls **INT** Low.

IORQ. *Input/Output Request* (input from CPU, active Low). **IORQ** is used in conjunction with **B/A**, **C/D**, **CE** and **RD** to transfer commands and data between the CPU and the SIO. When **CE**, **RD** and **IORQ** are all active, the channel selected by **B/A** transfers data to the CPU (a read operation). When **CE** and **IORQ** are active but **RD** is inactive, the channel selected by **B/A** is written to by the CPU with either data or control information as specified by **C/D**. If **IORQ** and **M1** are active simultane-

ously, the CPU is acknowledging an interrupt and the SIO automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

M1. *Machine Cycle* (input from Z-80 CPU, active Low). When **M1** is active and **RD** is also active, the Z-80 CPU is fetching an instruction from memory; when **M1** is active while **IORQ** is active, the SIO accepts **M1** and **IORQ** as an interrupt acknowledge if the SIO is the highest priority device that has interrupted the Z-80 CPU.

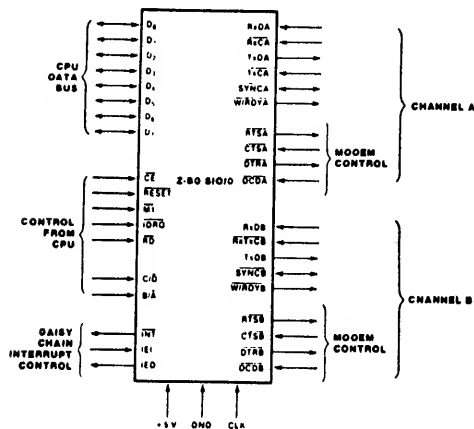
RxCA, RxCB. *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of **RxC**. The Receive Clocks may be 1, 16, 32 or 64 times the data rate in asynchronous modes. These clocks may be driven by the Z-80 CTC Counter Timer Circuit for programmable baud rate generation. Both inputs are Schmitt-trigger buffered (no noise level margin is specified).

In the Z-80 SIO/O bonding option, **RxCB** is bonded together with **TxCB**.

RD. *Read Cycle Status* (input from CPU, active Low). If **RD** is active, a memory or I/O read operation is in progress. **RD** is used with **B/A**, **CE** and **IORQ** to transfer data from the SIO to the CPU.

RxDA, RxDB. *Receive Data* (inputs, active High). Serial data at TTL levels.

RESET. *Reset* (input, active Low). A Low **RESET** disables both receivers and transmitters, forces **TxDA** and **TxDB** marking, forces the modem controls High and disables all interrupts. The control registers must be



Pin Description
(Continued)

rewritten after the SIO is reset and before data is transmitted or received.

RTSA, RTSB. *Request To Send* (outputs, active Low). When the RTS bit in Write Register 5 (Figure 14) is set, the RTS output goes Low. When the RTS bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the RTS pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

SYNCA, SYNCB. *Synchronization* (inputs/outputs, active Low). These pins can act either as inputs or outputs. In the asynchronous receive mode, they are inputs similar to $\overline{\text{CTS}}$ and $\overline{\text{DCD}}$. In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 (Figure 13), but have no other function. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved, SYNC must be driven Low on the second rising edge of $\overline{\text{RxC}}$ after that rising edge of $\overline{\text{RxC}}$ on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the SYNC input. Once SYNC is forced Low, it should be kept Low until the CPU informs the external synchronization detect logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of $\overline{\text{RxC}}$ that immediately precedes the falling edge of SYNC in the External Sync mode.

In the internal synchronization mode (Monosync and Bisync), these pins act as outputs that are active during the part of the receive clock ($\overline{\text{RxC}}$) cycle in which sync characters are recognized. The sync condition is not latched, so these outputs are active each time a sync pattern is recognized, regardless of character boundaries.

In the Z-80 SIO/2 bonding option, $\overline{\text{SYNCB}}$ is omitted.

TxCA, TxCB. *Transmitter Clocks* (inputs). In asynchronous modes, the Transmitter Clocks may be 1, 16, 32 or 64 times the data rate; however, the clock multiplier for the transmitter and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements (no noise level margin is specified). Transmitter Clocks may be driven by the Z-80 CTC Counter Timer Circuit for programmable baud rate generation.

In the Z-80 SIO/0 bonding option, $\overline{\text{TxCB}}$ is bonded together with $\overline{\text{RxCB}}$.

TxDA, TxDB. *Transmit Data* (outputs, active High). Serial data at TTL levels. TxD changes from the falling edge of $\overline{\text{TxC}}$.

W/RDYA, W/RDYB. *Wait/Ready A, Wait/Ready B* (outputs, open drain when programmed for Wait function, driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the SIO data rate. The reset state is open drain.

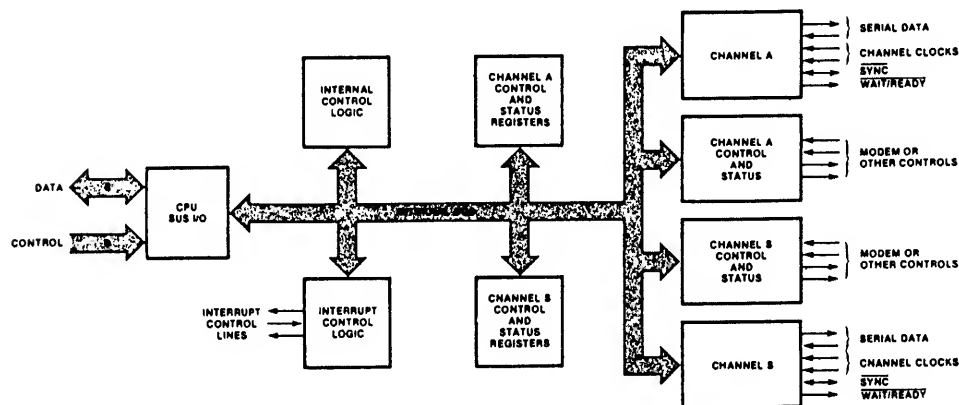


Figure 7. Block Diagram

Functional Description

The functional capabilities of the Z-80 SIO can be described from two different points of view: as a data communications device, it transmits and receives serial data in a wide variety of data-communication protocols; as a Z-80 family peripheral, it interacts with the Z-80 CPU and other peripheral circuits, sharing the data, address and control buses, as well as being a part of the Z-80 interrupt structure. As a peripheral to other microprocessors,

the SIO offers valuable features such as non-vectored interrupts, polling and simple hand-shake capability.

Figure 8 illustrates the conventional devices that the SIO replaces.

The first part of the following discussion covers SIO data-communication capabilities; the second part describes interactions between the CPU and the SIO.

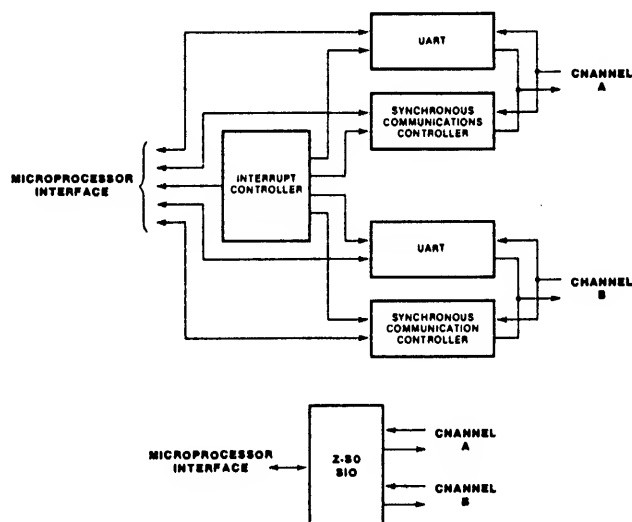


Figure 8. Conventional Devices Replaced by the Z-80 SIO

Data Communication Capabilities

The SIO provides two independent full-duplex channels that can be programmed for use in any common asynchronous or synchronous data-communication protocol. Figure 9 illustrates some of these protocols. The following is a short description of them. A more detailed explanation of these modes can be found in the *Z-80 SIO Technical Manual*.

Asynchronous Modes. Transmission and reception can be done independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in Figure 5). If the Low does not persist—as in the case of a transient—the character assembly process is not started.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occurred. Vectored

interrupts allow fast servicing of error conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The SIO does not require symmetric transmit and receive clock signals—a feature that allows it to be used with a Z-80 CTC or many other clock sources. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32 or 1/64 of the clock rate supplied to the receive and transmit clock inputs.

In asynchronous modes, the SYNC pin may be programmed as an input that can be used for functions such as monitoring a ring indicator.

Synchronous Modes. The SIO supports both byte-oriented and bit-oriented synchronous communication.

Synchronous byte-oriented protocols can be handled in several modes that allow character synchronization with an 8-bit sync character (Monosync), any 16-bit sync pattern (Bisync), or with an external sync signal. Leading sync

**Data
Communi-
cation
Capabilities**
(Continued)

characters can be removed without interrupting the CPU.

Five-, six- or seven-bit sync characters are detected with 8- or 16-bit patterns in the SIO by overlapping the larger pattern across multiple in-coming sync characters, as shown in Figure 10.

CRC checking for synchronous byte-oriented modes is delayed by one character time so the CPU may disable CRC checking on specific characters. This permits implementation of protocols such as IBM Bisync.

Both CRC-16 ($X^{16} + X^{15} + X^2 + 1$) and CCITT ($X^{16} + X^{12} + X^5 + 1$) error checking polynomials are supported. In all non-SDLC modes, the CRC generator is initialized to 0's; in SDLC modes, it is initialized to 1's. The SIO can be used for interfacing to peripherals such as hard-sectored floppy disk, but it cannot generate or check CRC for IBM-compatible soft-sectored disks. The SIO also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows very high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there is no data or CRC to send in synchronous modes, the transmitter inserts 8- or 16-bit sync characters regardless of the programmed character length.

The SIO supports synchronous bit-oriented protocols such as SDLC and HDLC by performing automatic flag sending, zero insertion and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message the SIO automatically transmits the CRC and trailing flag when the transmit buffer becomes empty. If a transmit

underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. One to eight bits per character can be sent, which allows reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically synchronizes on the leading flag of a frame in SDLC or HDLC, and provides a synchronization signal on the SYNC pin; an interrupt can also be programmed. The receiver can be programmed to search for frames addressed by a single byte to only a specified user-selected address or to a global broadcast address. In this mode, frames that do not match either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For transmitting data, an interrupt on the first received character or on every character can be selected. The receiver automatically deletes all zeroes inserted by the transmitter during character assembly. It also calculates and automatically checks the CRC to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers.

The SIO can be conveniently used under DMA control to provide high-speed reception or transmission. In reception, for example, the SIO can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SIO then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received.

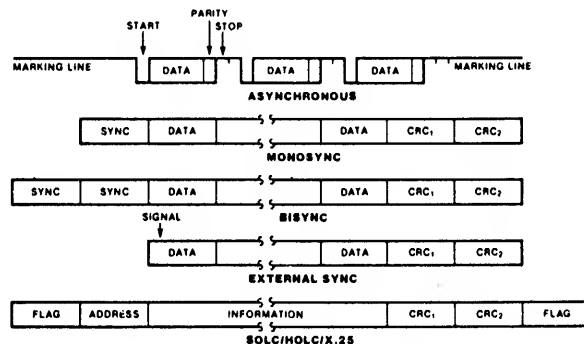


Figure 9. Some Z-80 SIO Protocols

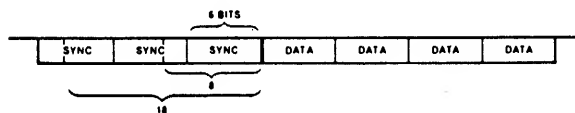


Figure 10.

2042-0108, 0109

I/O Interface Capabilities

The SIO offers the choice of polling, interrupt (vectored or non-vectored) and block-transfer modes to transfer data, status and control information to and from the CPU. The block-transfer mode can also be implemented under DMA control.

Polling. Two status registers are updated at appropriate times for each function being performed (for example, CRC error-status valid at the end of a message). When the CPU is operated in a polling fashion, one of the SIO's two status registers is used to indicate whether the SIO has some data or needs some data. Depending on the contents of this register, the CPU will either write data, read data, or just go on. Two bits in the register indicate that a data transfer is needed. In addition, error and other conditions are indicated. The second status register (special receive conditions) does not have to be read in a polling sequence, until a character has been received. All interrupt modes are disabled when operating the device in a polled environment.

Interrupts. The SIO has an elaborate interrupt scheme to provide fast interrupt service in real-time applications. A control register and a status register in Channel B contain the interrupt vector. When programmed to do so, the SIO can modify three bits of the interrupt vector in the status register so that it points directly to one of eight interrupt service routines in memory, thereby servicing conditions in both channels and eliminating most of the needs for a status-analysis routine.

Transmit interrupts, receive interrupts and external/status interrupts are the main sources of interrupts. Each interrupt source is enabled under program control, with Channel A having a higher priority than Channel B, and with receive, transmit and external/status interrupts prioritized in that order within each channel. When the transmit interrupt is enabled, the

CPU is interrupted by the transmit buffer becoming empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) The receiver can interrupt the CPU in one of two ways:

- Interrupt on first received character
- Interrupt on all received characters

Interrupt-on-first-received-character is typically used with the block-transfer mode. Interrupt-on-all-received-characters has the option of modifying the interrupt vector in the event of a parity error. Both of these interrupt modes will also interrupt under special receive conditions on a character or message basis (end-of-frame interrupt in SDLC, for example). This means that the special-receive condition can cause an interrupt only if the interrupt-on-first-received-character or interrupt-on-all-received-characters mode is selected. In interrupt-on-first-received-character, an interrupt can occur from special-receive conditions (except parity error) after the first-received-character interrupt (example: receive-overflow interrupt).

The main function of the external/status interrupt is to monitor the signal transitions of the Clear To Send (CTS), Data Carrier Detect (DCD) and Synchronization (SYNC) pins (Figures 1 through 6). In addition, an external/status interrupt is also caused by a CRC-sending condition or by the detection of a break sequence (asynchronous mode) or abort sequence (SDLC mode) in the data stream. The interrupt caused by the break/abort sequence allows the SIO to interrupt when the break/abort sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the break/abort condition in external logic.

I/O Interface Capabilities

In a Z-80 CPU environment (Figure 11), SIO interrupt vectoring is "automatic": the SIO passes its internally-modifiable 8-bit interrupt vector to the CPU, which adds an additional 8 bits from its interrupt-vector (I) register to form the memory address of the interrupt-routine table. This table contains the address of the beginning of the interrupt routine itself. The process entails an indirect transfer of CPU control to the interrupt routine, so that the next instruction executed after an interrupt acknowledge by the CPU is the first instruction of the interrupt routine itself.

CPU/DMA Block Transfer. The SIO's block-transfer mode accommodates both CPU block transfers and DMA controllers (Z-80 DMA or other designs). The block-transfer mode uses the Wait/Ready output signal, which is selected with three bits in an internal control register. The Wait/Ready output signal can be programmed as a WAIT line in the CPU block-transfer mode or as a READY line in the DMA block-transfer mode.

To a DMA controller, the SIO READY output indicates that the SIO is ready to transfer data to or from memory. To the CPU, the WAIT output indicates that the SIO is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

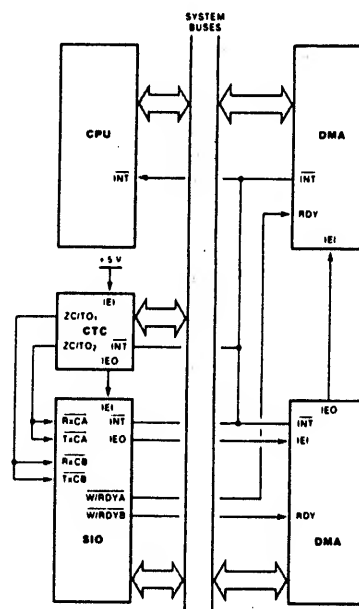


Figure 11. Typical Z-80 Environment

Internal Structure

The internal structure of the device includes a Z-80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains its own set of control and status (write and read) registers, and control and status logic that provides the interface to modems or other external devices.

The registers for each channel are designated as follows:

- WR0-WR7 — Write Registers 0 through 7
- RR0-RR2 — Read Registers 0 through 2

The register group includes five 8-bit control registers, two sync-character registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B that may be read through another 8-bit register (Read Register 2) in Channel B. The bit assignment and functional grouping of each register is configured to simplify and organize the programming process. Table 1 lists the functions assigned to each read or write register.

Read Register Functions

RR0	Transmit/Receive buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)

Write Register Functions

WR0	Register pointers, CRC initialize, initialization commands for the various modes, etc.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync character or SDLC address field
WR7	Sync character or SDLC flag

2032-0127

Internal Structure
(Continued)

The logic for both channels provides formats, synchronization and validation for data transferred to and from the channel interface. The modem control inputs, Clear To Send (CTS) and Data Carrier Detect (DCD), are monitored by the external control and status logic under program control. All external control-and-status-logic signals are general-purpose in nature and can be used for functions other than modem control.

Data Path. The transmit and receive data path illustrated for Channel A in Figure 12 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the

CPU to service an interrupt at the beginning of a block of high-speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode and—in asynchronous modes—the character length.

The transmitter has an 8-bit transmit data buffer register that is loaded from the internal data bus, and a 20-bit transmit shift register that can be loaded from the sync-character buffers or from the transmit data register. Depending on the operational mode, outgoing data is routed through one of four main paths before it is transmitted from the Transmit Data output (TxD).

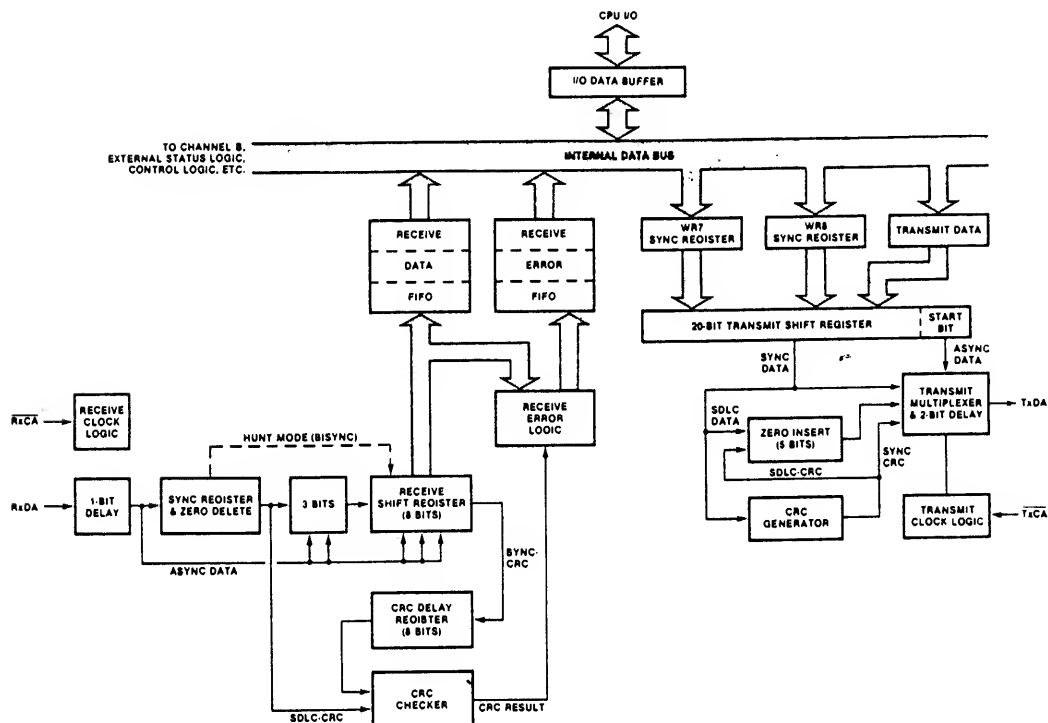


Figure 12. Transmit and Receive Data Path (Channel A)

Programming The system program first issues a series of commands that initialize the basic mode of operation and then other commands that qualify conditions within the selected mode. For example, the asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first; then the interrupt mode; and finally, receiver or transmitter enable.

Both channels contain registers that must be programmed via the system program prior to operation. The channel-select input (B/\bar{A}) and the control/data input (C/\bar{D}) are the command-structure addressing controls, and are normally controlled by the CPU address bus. Figures 15 and 16 illustrate the timing relationships for programming the write registers and transferring data and status.

Read Registers. The SIO contains three read registers for Channel B and two read registers for Channel A (RR0-RR2 in Figure 13) that can be read to obtain the status information; RR2 contains the internally-modifiable interrupt vector and is only in the Channel B register set. The status information includes error conditions, interrupt vector and standard communications-interface signals.

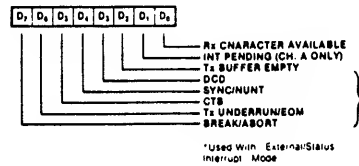
To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing a read instruction, the contents of the addressed read register can be read by the CPU.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Condition interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

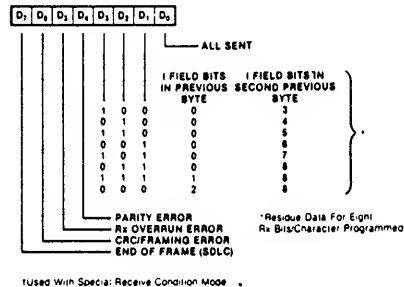
Write Registers. The SIO contains eight write registers for Channel B and seven write registers for Channel A (WR0-WR7 in Figure 14) that are programmed separately to configure the functional personality of the channels; WR2 contains the interrupt vector for both channels and is only in the Channel B register set. With the exception of WR0, programming the write registers requires two bytes. The first byte is to WR0 and contains three bits (D_0 - D_2) that point to the selected register; the second byte is the actual control word that is written into the register to configure the SIO.

WR0 is a special case in that all of the basic commands can be written to it with a single byte. Reset (internal or external) initializes the pointer bits D_0 - D_2 to point to WR0. This implies that a channel reset must not be combined with the pointing to any register.

READ REGISTER 0



READ REGISTER 1†



READ REGISTER 2

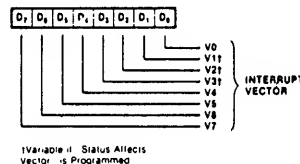
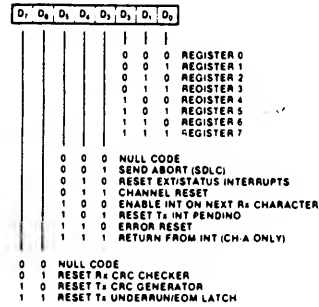


Figure 13. Read Register Bit Functions

Programming
(Continued)

WRITE REGISTER 0



WRITE REGISTER 1

Timing

The SIO must have the same clock as the CPU (same phase and frequency relationship, not necessarily the same driver).

Read Cycle. The timing signals generated by a Z-80 CPU input instruction to read a data or status byte from the SIO are illustrated in Figure 15.

Write Cycle. Figure 16 illustrates the timing and data signals generated by a Z-80 CPU output instruction to write a data or control byte into the SIO.

Interrupt-Acknowledge Cycle. After receiving an interrupt-request signal from an SIO (INT pulled Low), the Z-80 CPU sends an interrupt-acknowledge sequence (\overline{MI} Low, and \overline{IORQ} Low a few cycles later) as in Figure 17.

The SIO contains an internal daisy-chained interrupt structure for prioritizing nested interrupts for the various functions of its two channels, and this structure can be used within an external user-defined daisy chain that prioritizes several peripheral circuits.

The IEI of the highest-priority device is terminated High. A device that has an interrupt pending or under service forces its IEO Low. For devices with no interrupt pending or under service, $IEO = IEI$.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while \overline{MI} is Low. When \overline{IORQ} is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its

internal interrupt-under-service latch.

Return From Interrupt Cycle. Figure 18 illustrates the return from interrupt cycle. Normally, the Z-80 CPU issues a RETI (Return From Interrupt) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch in the SIO to terminate the interrupt that has just been processed. This is accomplished by manipulating the daisy chain in the following way.

The normal daisy-chain operation can be used to detect a pending interrupt; however, it cannot distinguish between an interrupt under service and a pending unacknowledged interrupt of a higher priority. Whenever "ED" is decoded, the daisy chain is modified by forcing High the IEO of any interrupt that has not yet been acknowledged. Thus the daisy chain identifies the device presently under service as the only one with an IEI High and an IEO Low. If the next opcode byte is "4D," the interrupt-under-service latch is reset.

The ripple time of the interrupt daisy chain (both the High-to-Low and the Low-to-High transitions) limits the number of devices that can be placed in the daisy chain. Ripple time can be improved with carry-look-ahead, or by extending the interrupt-acknowledge cycle. For further information about techniques for increasing the number of daisy-chained devices, refer to the Z-80 CPU Product Specification.

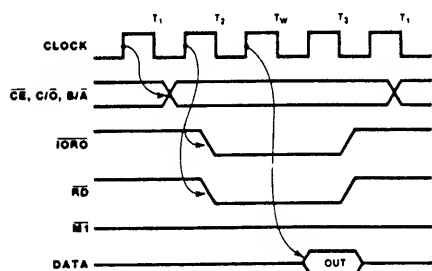


Figure 15. Read Cycle

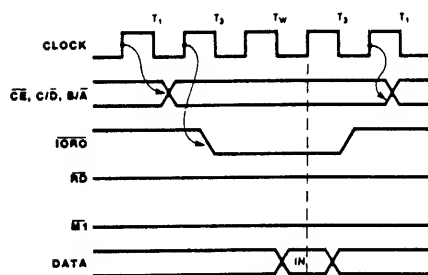


Figure 16. Write Cycle

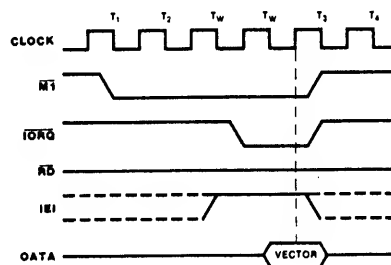


Figure 17. Interrupt Acknowledge Cycle

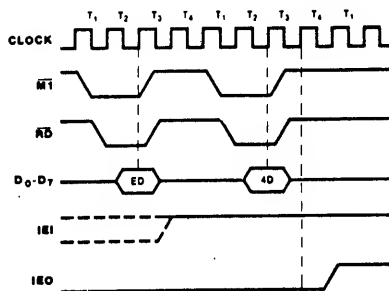


Figure 18. Return from Interrupt Cycle

2044-008, 009, 010, 011

Absolute Maximum Ratings

Voltages on all inputs and outputs with respect to GND. -0.3 V to +7.0 V

Operating Ambient Temperature As Specified in Ordering Information

Storage Temperature -65°C to +150°C

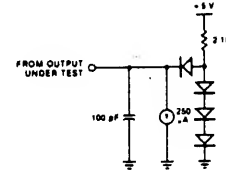
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Test Conditions

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are:

- 0°C to +70°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -40°C to +85°C,
+4.75 V ≤ V_{CC} ≤ +5.25 V
- -55°C to +125°C,
+4.5 V ≤ V_{CC} ≤ +5.5 V

The product number for each operating temperature range may be found in the ordering information section.



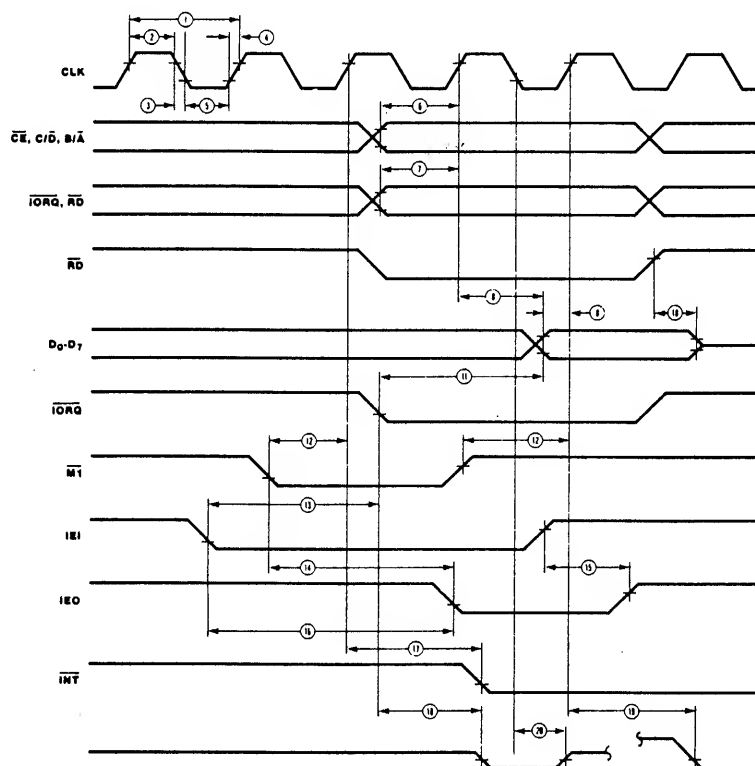
DC Characteristics	Symbol	Parameter	Min	Max	Unit	Test Condition
	V _{ILC}	Clock Input Low Voltage	-0.3	+0.45	V	
	V _{IHC}	Clock Input High Voltage	V _{CC} -0.6	+5.5	V	
	V _{IL}	Input Low Voltage	-0.3	+0.8	V	
	V _{IH}	Input High Voltage	+2.0	+5.5	V	
	V _{OL}	Output Low Voltage		+0.4	V	I _{OL} = 2.0 mA
	V _{OH}	Output High Voltage	+2.4		V	I _{OH} = -250 μA
	I _{LI}	Input Leakage Current	-10	+10	μA	0 < V _{IN} < V _{CC}
	I _Z	3-State Output/Data Bus Input Leakage Current	-10	+10	μA	0 < V _{IN} < V _{CC}
	I _{LSY}	SYN _C Pin Leakage Current	-40	+10	μA	0 < V _{IN} < V _{CC}
	I _{CC}	Power Supply Current		100	mA	

Over specified temperature and voltage range

Capacitance	Symbol	Parameter	Min	Max	Unit	Test Condition
	C	Clock Capacitance	40	pF	Unmeasured	
	C _{IN}	Input Capacitance	5	pF	pins returned	
	C _{OUT}	Output Capacitance	10	pF	to ground	

Over specified temperature range, f = 1MHz

AC
Electrical
Character-
istics

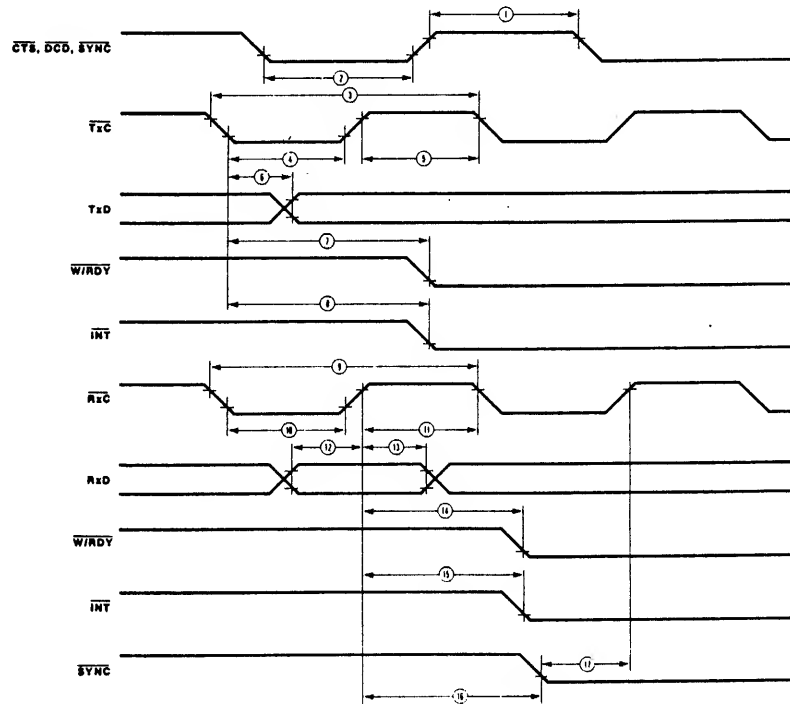


Number	Symbol	Parameter	Z-80 SIO		Z-80A SIO		Z-80B SIO		Unit
			Min	Max	Min	Max	Min	Max	
1	T _c C	Clock Cycle Time	400	4000	250	4000	165	4000	ns
2	T _w Ch	Clock Width (High)	170	2000	105	2000	70	2000	ns
3	T _f C	Clock Fall Time		30		30		15	ns
4	T _r C	Clock Rise Time		30		30		15	ns
5	T _w Cl	Clock Width (Low)	170	2000	105	2000	70	2000	ns
6	T _s AD(C)	CE, C/D, B/A to Clock 1 Setup Time	160		145		60		ns
7	T _s CS(C)	IORQ, RD to Clock 1 Setup Time	240		115		60		ns
8	T _d C(DO)	Clock 1 to Data Out Delay		240		220		150	ns
9	T _s DI(C)	Data In to Clock 1 Setup (Write or M1 Cycle)	50		50		30		ns
10	T _d RD(DOz)	RD 1 to Data Out Float Delay		230		110		90	ns
11	T _d IO(DOI)	IORQ 1 to Data Out Delay (INTACK Cycle)		340		160		100	ns
12	T _s M1(C)	M1 to Clock 1 Setup Time	210		90		75		ns
13	T _s IEI(IO)	IEI to IORQ 1 Setup Time (INTACK Cycle)	200		140		120		ns
14	T _d M1(IEO)	M1 1 to IEO 1 Delay (interrupt before M1)		300		190		160	ns
15	T _d IEI(IEOr)	IEI 1 to IEO 1 Delay (after ED decode)		150		100		70	ns
16	T _d IEI(IEOf)	IEI 1 to IEO 1 Delay		150		100		70	ns
17	T _d C(INT)	Clock 1 to INT 1 Delay		200		200		150	ns
18	T _d IO(W/RWf)	IORQ 1 or CE 1 to W/RDY 1 Delay Wait Mode)		300		210		175	ns
19	T _d C(W/RR)	Clock 1 to W/RDY 1 Delay (Ready Mode)		120		120		100	ns
20	T _d C(W/RWz)	Clock 1 to W/RDY Float Delay (Wait Mode)		150		130		110	ns
21	Th	Any unspecified Hold when Setup is specified	0		0		0		ns

2044-012

ZILOG DATA
Z80 SIO

AC
Electrical
Character-
istics
(Continued)



Number	Symbol	Parameter	Z-80 SIO		Z-80A SIO		Z-80B SIO		Unit
			Min	Max	Min	Max	Min	Max	
1	TwPh	Pulse Width (High)	200		200		200		ns
2	TwPl	Pulse Width (Low)	200		200		200		ns
3	TcTx C	$\overline{\text{Tx C}}$ Cycle Time	400	∞	400	∞	330	∞	ns
4	TwTx C1	$\overline{\text{Tx C}}$ Width (Low)	180	∞	180	∞	100	∞	ns
5	TwTx Ch	$\overline{\text{Tx C}}$ Width (High)	180	∞	180	∞	100	∞	ns
6	TdTx C(TxD)	$\overline{\text{Tx C}}$ 1 to Tx D Delay (x1 Mode)		400		300		220	ns
7	TdTx C(W/RRt)	$\overline{\text{Tx C}}$ 1 to $\overline{\text{W/RDY}}$ 1 Delay (Ready Mode)	5	9	5	9	5	9	Clk Periods*
8	TdTx C(INT)	$\overline{\text{Tx C}}$ 1 to $\overline{\text{INT}}$ 1 Delay	5	9	5	9	5	9	Clk Periods*
9	TcRx C	$\overline{\text{Rx C}}$ Cycle Time	400	∞	400	∞	330	∞	ns
10	TwRx C1	$\overline{\text{Rx C}}$ Width (Low)	180	∞	180	∞	100	∞	ns
11	TwRx Ch	$\overline{\text{Rx C}}$ Width (High)	180	∞	180	∞	100	∞	ns
12	TsRx D(RxC)	RxD to $\overline{\text{Rx C}}$ 1 Setup Time (x1 Mode)	0		0		0		ns
13	ThRx D(RxC)	$\overline{\text{Rx C}}$ 1 to RxD Hold Time (x1 Mode)	140		140		100		ns
14	TdRx C(W/RRt)	$\overline{\text{Rx C}}$ 1 to $\overline{\text{W/RDY}}$ 1 Delay (Ready Mode)	10	13	10	13	10	13	Clk Periods*
15	TdRx C(INT)	$\overline{\text{Rx C}}$ 1 to $\overline{\text{INT}}$ 1 Delay	10	13	10	13	10	13	Clk Periods*
16	TdRx C(SYNC)	$\overline{\text{Rx C}}$ 1 to $\overline{\text{SYNC}}$ 1 Delay (Output Modes)	4	7	4	7	4	7	Clk Periods*
17	TsSYNC(RxC)	$\overline{\text{SYNC}}$ 1 to $\overline{\text{Rx C}}$ 1 Setup (External Sync Modes)	-100		-100		100		ns

In all modes, the System Clock rate must be at least five times the maximum data rate.
RESET must be active a minimum of one complete Clock Cycle.
*System Clock

Z044 013

ZILOG DATA
Z80 SIO

Ordering Information	Product Number	Package/Temp	Speed	Description	Product Number	Package/Temp	Speed	Description
	Z8440	CE,CM	2.5 MHz	Z80 SIO/0 (40-pin)	Z8441A	DE,DS	4.0 MHz	Z80A SIO/1 (40-pin)
	Z8440	CMB,CS	2.5 MHz	Same as above	Z8441A	PE,PS	4.0 MHz	Same as above
	Z8440	DE,DS	2.5 MHz	Same as above	Z8441B	CE,CM	6.0 MHz	Z80B SIO/1 (40-pin)
	Z8440	PE,PS	2.5 MHz	Same as above	Z8441B	CMB,CS	6.0 MHz	Same as above
	Z8440A	CE,CM	4.0 MHz	Z80A SIO/0 (40-pin)	Z8441B	DE,DS	6.0 MHz	Same as above
	Z8440A	CMB,CS	4.0 MHz	Same as above	Z8441B	PE,PS	6.0 MHz	Same as above
	Z8440A	DE,DS	4.0 MHz	Same as above	Z8442	CE,CM	2.5 MHz	Z80 SIO/2 (40-pin)
	Z8440A	PE,PS	4.0 MHz	Same as above	Z8442	CMB,CS	2.5 MHz	Same as above
	Z8440B	CE,CM	6.0 MHz	Z80B SIO/0 (40-pin)	Z8442	DE,DS	2.5 MHz	Same as above
	Z8440B	CMB,CS	6.0 MHz	Same as above	Z8442	PE,PS	2.5 MHz	Same as above
	Z8440B	DE,DS	6.0 MHz	Same as above	Z8442A	CE,CM	4.0 MHz	Z80A SIO/2 (40-pin)
	Z8440B	PE,PS	6.0 MHz	Same as above	Z8442A	CMB,CS	4.0 MHz	Same as above
	Z8441	CE,CM	2.5 MHz	Z80 SIO/1 (40-pin)	Z8442A	DE,DS	4.0 MHz	Same as above
	Z8441	CMB,CS	2.5 MHz	Same as above	Z8442A	PE,PS	4.0 MHz	Same as above
	Z8441	DE,DS	2.5 MHz	Same as above	Z8442B	CE,CM	6.0 MHz	Z80B SIO/2 (40-pin)
	Z8441	PE,PS	2.5 MHz	Same as above	Z8442B	CMB,CS	6.0 MHz	Same as above
	Z8441A	CE,CM	4.0 MHz	Z80A SIO/1 (40-pin)	Z8442B	DE,DS	6.0 MHz	Same as above
	Z8441A	CMB,CS	4.0 MHz	Same as above	Z8442B	PE,PS	6.0 MHz	Same as above

NOTES: C = Ceramic, D = Cerdip, P = Plastic; E = -40°C to +85°C, M = -55°C to +125°C, MB = -55°C to +125°C with MIL-STD-883 with Class B processing, S = 0°C to +70°C.

00-2042-A

ZILOG DATA
Z80 SIO

THEORY OF OPERATION

THEORY OF OPERATION

CENTRAL PROCESSOR

CLOCK GENERATOR:

All the system clocks with the exception of the baud clock and the video dot clock are generated from a master oscillator operating at 20 Mhz.

The 20 Mhz clock is scaled by the divide-by-5 section of decade counter U-12 to provide 4 Mhz for use in the floppy disk data separator. The 2 Mhz clock for the disk controller is generated from the 4 Mhz clock by the remaining divide by two sections of U-12.

The 2.5 Mhz processor clock is generated by dividing the master 20 Mhz clock by 8 with binary counter U-10. The output of the third stage is buffered by inverter U-9 and transistor Q-1.

The column address strobe "CAS", and the address multiplexer control "MUXC", are derived from the 20 Mhz clock. When memory request "MREQB" is low and refresh "RFSHB" is high, generation of "CAS" and "MUXC" is enabled. "RFSHB" disables the generation of "CAS" and "MUXC" by holding shift register U-11 reset. This is done to take advantage of the low power row address strobe "RAS" only refresh mode of the 16 K dynamic RAMs.

RESET CONTROLLER:

Two types of reset take place on the board. Power on reset is detected and conditioned by part of hex schmitt inverter U-108. The pushbutton reset is also conditioned by a part of hex schmitt inverter U-108. The "D" type flip flop U-26 synchronizes the pushbutton reset with machine cycle one "M1" from the processor. The output of the flip flop triggers a 12 microsecond one shot U-27. Power on reset and pushbutton reset are or ed together by U-28 and inverted by U-29 for use by the processor. The reset pulse is negative or ed with "M1" by U-45 to generate a reset for the Z80 family programmable I/O devices.

BUS BUFFERING:

Octal buffer U-78 buffers the control signals generated by the processor for use though-out the system. Quad transceivers U-30 and U-31 mediate data transfers to and from memory. U-79 and U-45 control the direction of the data bus transceivers. During a memory read the data transceivers allow data from memory through to the processor, otherwise the processor always drives memory. Octal buffer U-81 drives the lower 8 bits of the address bus. The octal latch U-35 serves a dual function, as well as buffering the upper 8 bits of the address bus, the latch holds the address bus stable during the active portion of the "MREQ" cycle the Z80 microprocessor allows the address bus to change.

READ ONLY MEMORY:

The board can accommodate up to 4K of 2716 ROM.

U-64 RESIDES FROM 0000 HEX TO 07FF HEX
U-63 RESIDES FROM 0800 HEX TO 0FFF HEX

The description of the bank switching technique will be covered with the 64 K RAM theory of operation.

PORT ADDRESS DECODING:

Octal decoder U-88 is used to select the appropriate I/O device based on the binary value of the address bits A2, A3, & A4. When A7 is low and "M1R" is high, a low on "IORQ" will cause the appropriate output of the decoder to go low, selecting the I/O device for a read or write operation.

THEORY OF OPERATION

DISK TRANSFER SYNCHRONIZATION:

In order to successfully execute the high speed data transfers between the processor and the disk controller; the fast Z80 non maskable interrupt "NMI" response was employed. During reads and writes to and from the disk controller, the data at memory location 66 hex is retrieved and stored. This location is overwritten with a RETURN instruction. After this setup is accomplished the processor executes a HALT instruction. When the processor is in a HALT condition, a DATA REQUEST (DRQ) or an INTERRUPT REQUEST (IRQ) from the disk controller will cause a non-maskable interrupt to be generated. The processor then executes the RETURN instruction at 66 hex and returns to transfer the data to or from the disk controller. When the 128 byte transfer is complete the old data is restored at location 66 hex and the processor resumes normal operation. This hardware assistance obviated the necessity for a DMA device by eliminating the disk controller "DRQ" status test.

CRT DISPLAY GENERATOR

VIDEO CLOCK GENERATION:

Three inverters from U-14 are used to generate the video dot clock. The 14.31818 Mhz dot clock is divided by 7 to develop the character clock. Synchronous binary counter U-50 is preloaded with a binary 9 at each top count to accomplish the divide by 7 function. The character clock is divided by 128 by the 8 bit binary counter U-53 to develop the scan clock. In the process of developing the scan clock the intermediate outputs of U-53 develop part of the character address for the video RAM. Decade counter U-52 divides the scan clock by 10, simultaneously developing the line clock and the vertical component of the character matrix address. U-49 and part of U-51 work in conjunction to generate the frame clock and the line address for the video RAM. The two devices divide the line clock by 26 to generate the 60 hz frame clock. The second half of U-49 divides the frame clock by 16 to develop the 4 hz blink clock.

VIDEO RAM ADDRESSING:

Multiplexers U-67, U-69 and U-70 select the source of the addresses for the video RAM. If the processor is doing a read or write to video RAM "CRTCE" (CRT memory access enable) will go low. When "CRTCE" goes low, the address from the processor is selected instead of the address generated by the counter chain. This gives the processor access to the video RAM for read out write operations. U-68 maps the 12 bit address developed by the counter chain into the 2 K byte video RAM.

SYNC GENERATION:

Horizontal sync is generated by decoding the 80th count of the character counter U-53.

The vertical sync is generated between counts 24 and 26 of the line counter.

CPU ACCESS OF VIDEO RAM:

During read or write operations involving the video RAM and the CPU, "CRTCE" will go low. When "CRTCE" goes low the processor address bus is selected by multiplexers U-69 - U-70 as the address source for the video RAM. A low on "CRTCE" is also used as a term in the direction control logic for data bus access. Decoder U-80 controls the direction and activity of transceivers U-82 and U-83. During a processor read operation, data from the video RAM at the specified address is allowed onto the processor data bus. During a processor write operation, data from the processor is written to the video RAM at the specified address.

VIDEO GENERATION:

While in the display mode, ASCII data from the video RAM and scan address data from decade counter U-52 are used to select the proper dot patterns from the character generator U-92. The dot information from the character generator is sampled by hex "D" flip flop U-91 at the next character time. While the next character is being accessed, the previous dot pattern is multiplexed out of U-91 by multiplexer U-90. Multiplexer U-90 feeds the video driver U-117.

THEORY OF OPERATION

DISPLAY BLANKING:

The display is blanked during horizontal retrace, vertical retrace, CPU access, and decode of scan counts 8 & 9. Blanking is accomplished by disabling the character generator.

CRT RAM MEMORY ALLOCATION

The CRT RAM resides from 3000 hex to 3FFF hex. Each 80 character line on the screen is allocated 128 bytes in the CRT RAM. Listed below are the starting and ending addresses for each of the 24 rows in the CRT RAM (Assumes scroll register = 23 decimal).

ROW	0	3000 - 304F hex
ROW	1	3080 - 30CF hex
ROW	2	3100 - 314F hex
ROW	3	3180 - 31CF hex
ROW	4	3200 - 324F hex
ROW	5	3280 - 32CF hex
ROW	6	3300 - 334F hex
ROW	7	3380 - 33CF hex
ROW	8	3400 - 344F hex
ROW	9	3480 - 34CF hex
ROW	10	3500 - 354F hex
ROW	11	3580 - 35CF hex
ROW	12	3600 - 364F hex
ROW	13	3680 - 36CF hex
ROW	14	3700 - 374F hex
ROW	15	3780 - 37CF hex
ROW	16	3800 - 384F hex
ROW	17	3880 - 38CF hex
ROW	18	3900 - 394F hex
ROW	19	3980 - 39CF hex
ROW	20	3A00 - 3A4F hex
ROW	21	3A80 - 3ACF hex
ROW	22	3B00 - 3B4F hex
ROW	23	3B80 - 3BCF hex

The following example are character locations in the CRT memory.
(Assumes scroll register = 23 decimal)

ROW	COLUMN	MEMORY LOCATION
0	0	3000 (hex)
0	79	304F (hex)
1	0	3080 (hex)
1	79	30CF (hex)
10	0	3500 (hex)
10	79	354F (hex)
23	0	3B80 (hex)
23	79	3BCF (hex)

VIDEO SCROLLING

In order to eliminate the delay associated with software scrolling, hardware assistance was employed. Writing into the scroll register adds an offset to the line address developed by the line counter. For instance, an offset of zero puts the data at location 3000 hex (in the CRT memory) on the bottom row (row 23) of the screen. If the offset was one, the data at 3000 hex would be displayed on row 22. An offset of 23 (decimal) puts the data at location 3000 (hex) on row 0.

THEORY OF OPERATION

Scroll Register Contents	Memory location containing character displayed at Row 0, Column 0	Memory location containing character displayed at Row 23, Column 0
23 decimal	3000 hex	3B80 hex
22	3080	3B00
21	3100	3A80
20	3180	3A00
19	3200	3980
18	3280	3900
17	3300	3880
16	3380	3800
15	3400	3780
14	3480	3700
13	3500	3680
12	3580	3600
11	3600	3580
10	3680	3500
9	3700	3480
8	3780	3400
7	3800	3380
6	3880	3300
5	3900	3280
4	3980	3200
3	3A00	3180
2	3A80	3100
1	3B00	3080
0	3B80	3000

64 K RAM AND BANK SWITCHING

RAM ADDRESS MULTIPLEXING:

The address from the processor is multiplexed to the RAM array by multiplexers U-71 and U-72. During a memory access the row address is presented to the array first. After the row address is stable the decode of A15B and A14B gated by "MREQ", generates the proper row address strobe. The decode of A15B and A14B is accomplished by octal decoder U-62. Nand gate package U-80 gates the decoder outputs with "MREQ" to generate the "RAS" for the appropriate 16 K block. After the proper setup and hold time for the row address have been met, "MUXC" switches the column address on to the RAM array. After the setup block that received the "RAS". If the memory is being read, the data from the RAMs will be gated onto the data bus by transceivers U-73 and U-76. If the memory is being written to, data is routed from the processors data bus to the RAM array.

REFRESH:

During the refresh cycle, the Z-80 places the refresh address on the lower bits of the address bus. When this address is stable in the RAM array, the "RFSH" pin on the Z-80 goes low. The active low "RFSH" generates a "RAS" on all RAMS via nand gate packages U-77 and U-80. An active "RFSH" disables the generation of both "CAS" and "MUXC".

BANK SWITCHING:

Bit 7 of port 1C hex is the bank switch control. When the output is high, the ROMs and the CRT display appear in the lower 16K block. When bit 7 of port 1C hex is low, all the 64K RAM is available to the processor. Enabling of the CRT bank and the first 16K RAM bank are mutually exclusive. Data movement to or from one will not effect the other.

THEORY OF OPERATION

FLOPPY DISK CONTROLLER, SYSTEM PIO, AND CTC

FLOPPY DISK CONTROLLER:

The 1771 (U-109) performs all the control functions required to interface to a floppy disk drive. The only support required by the 1771 is external data separation, inverting data bus transceivers, head load timer, and buffering to and from the drive(s).

DATA SEPARATOR:

Presetable counter U-93 is used as a digital monostable with the timing reference developed by the system clock. Raw data coming from the disk drive is used to preload the counter. If the counter does not receive a data bit between clocks the counter in effect times out and presents the controller with a logic zero. If the counter receives data between clocks, the controller will see a logic one on its data input.

HEAD LOAD TIMING:

When the 1771 activates the head load output, monostable U-106 is triggered. The 1771 samples the "HLT" until a logic one is detected. At this time the head is assumed to be loaded and stable.

DATA BUS BUFFERING:

Inverting transceivers U-110 and U-119 adapt the 1771 to the non-inverted Z-80 data bus. During a read operation, data from the 1771 is allowed onto the processors data bus. Otherwise the processor's data bus always drives the 1771's data inputs.

CONTROL BUS BUFFERING:

U-118, part of U-47, and U-108 buffer the control, status and data to and from the 1771. In addition to buffering and isolation, U-108 and U-47 provide schmitt trigger characteristics for noise rejection.

CTC:

The Z80 CTC (Counter, Timer Controller) U99 resides at ports 18 hex through 1B hex.

SYSTEM PIO:

The system Z80 PIO resides at ports 1C hex through 1F hex. The "A" side of the system Z80 PIO controls the floppy disk drive select, bank switching, disk power switching, sensing keyboard data available (for polled keyboard applications), and on uncommitted user definable I/O bit. The bit allocations are as follows:

- BIT 0 = DVSEL 1
- BIT 1 = DVSEL 2
- BIT 2 = SIDE SELECT
- BIT 3 IS USED FOR KEYBOARD DATA AVAILABLE
- BIT 4 IS 8"/5" DISK SELECT
- BIT 5 ASSIGNED FOR FUTURE USE
- BIT 6 CONTROLS DISPLAY CHARACTER SET
- BIT 7 CONTROLS THE BANK SWITCHING (0=RAM)

The "B" side of the system Z80 PIO is devoted to the keyboard. The keyboard port is eight bits wide and is fully buffered.

THEORY OF OPERATION

GENERAL PURPOSE Z80 PIO AND Z80 SIO

The G.P. Z80 PIO U-101 provides the user with 16 bits of user definable input or output or a mix of input and output on nibble boundaries. The G.P. Z80PIO resides at ports 08 hex -0B hex. The PIO will support all modes of interrupt supported by the Z80.

SIO:

The Z80 SIO U-96 supports two full channels of serial I/O with the capability of supporting full RS-232 protocol on both channels. In addition, the A side of the Z80 SIO can provide clocks to synchronous modems or receive clocks from the modem. Channel A of the Z80 SIO can be configured to interface to a modem or a terminal.

BUAD RATE GENERATOR:

The COM 8116 U-97 provides the user with two programmable baud rate generators. Channel A baud rate resides at port 00 hex and is write only. Channel B baud rate resides at port 0C hex and is also write only.

THEORY OF OPERATION

NOTES

1 UNLESS OTHERWISE SPECIFIED
RESISTANCE VALUES ARE IN OHMS.
± 5%, 25M

CAPACITANCE VALUES ARE IN
MICROFARADS, ± 20%, 50P

2 POWER DISTRIBUTION TABLE

REF DESIGNATIONS	GND	+5	+12	-12	-5
U1, 8, 17, 24, 37, 44, 54, 61	16	9	8		1
U7, 12, 14, 16, 25, 26, 28, 29, 34, 36, 45, 49, 51, 52, 53, 74, 77, 78, 79, 82, 83, 89, 94, 95, 98, 100, 102, 103, 104, 108, 110, 112, 114, 115, 119	7	14			
U13	12	3			
U27, 48, 50, 62, 65, 73, 75, 76, 80, 84, 90, 91, 93, 106, 107	8	16			
U35, 47, 81	10	20			
U46	29	11			
U63, 64, 92	12	24			
U84, 87	9	15			
U96	31	9			
U97	11	2			
U99	5	24			
U101, 105	11	26			
U109	20	21	40		1
U11, 113	7		14		1

3 REFERENCE DESIGNATIONS

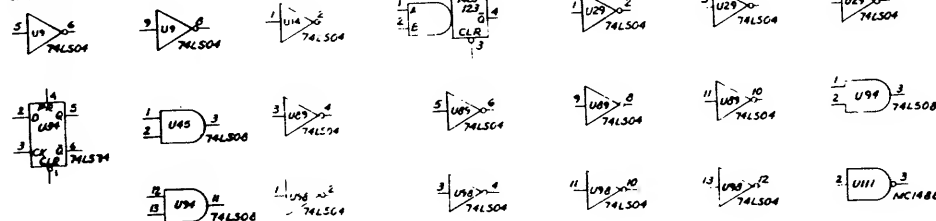
LAST USED	NOT USED
C195	C137
E3	
J11	
R69	R34, 49
S1	
U19	U63
Q1	
VR2	

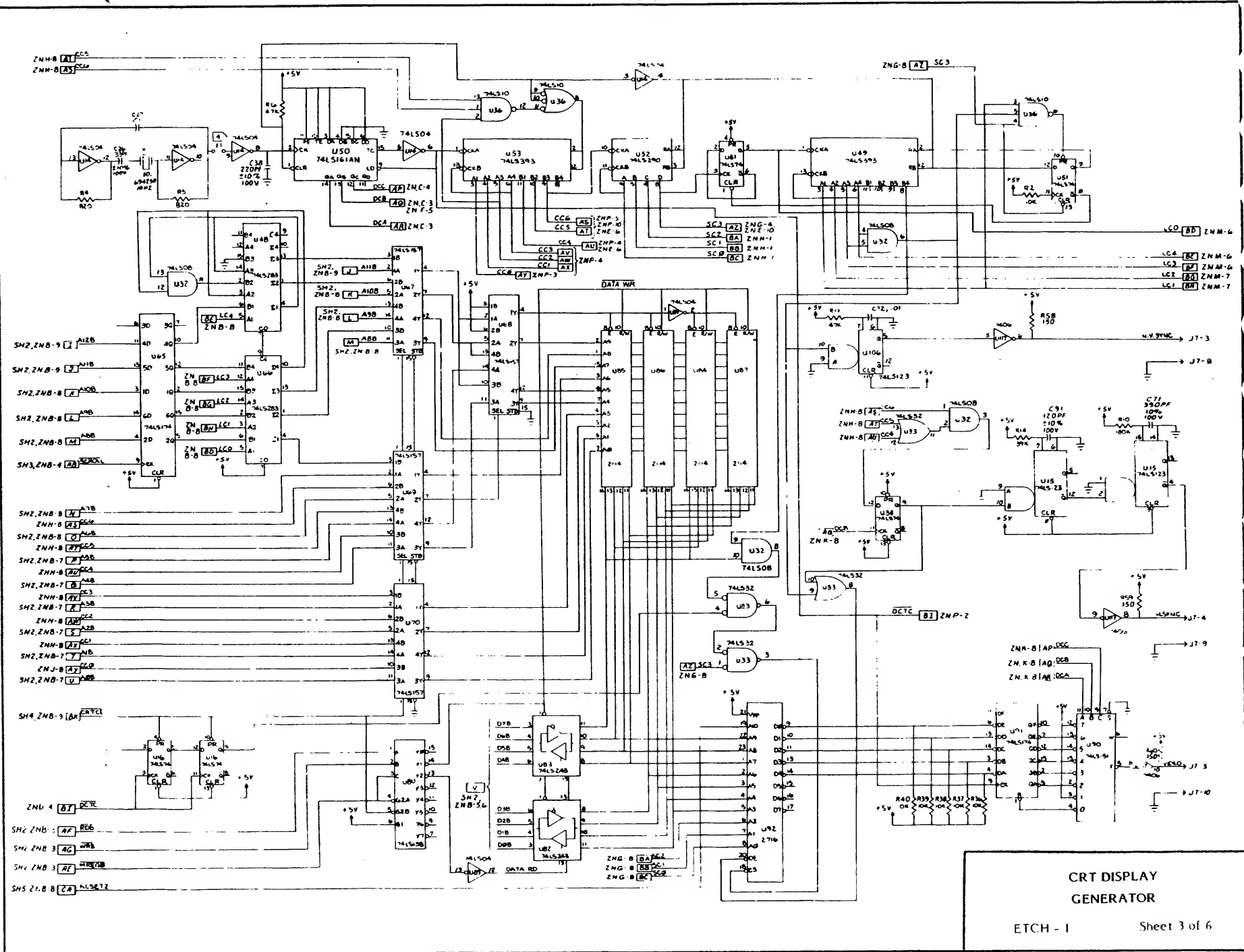
4 FOR NORMAL OPERATION SHUNTS TO BE INSTALLED IN THE FOLLOWING POSITIONS

REF DESIG	BETWEEN PINS
E1	1, 2
E2	1, 2
J1	7, 12, 11, 12, 15, 16 14, 20, 23, 124 27, 28, 31, 32 35, 36
J11	3, 14, 7, 18

5 LAST INTERCONNECT LETTER USED "CD"

6 SPARE GATES:



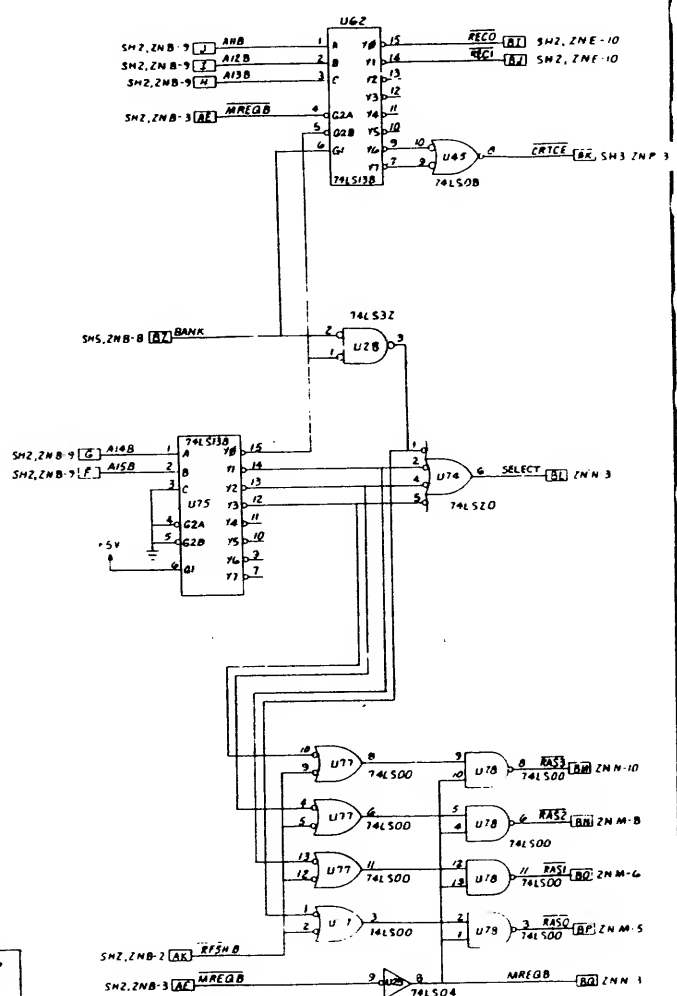
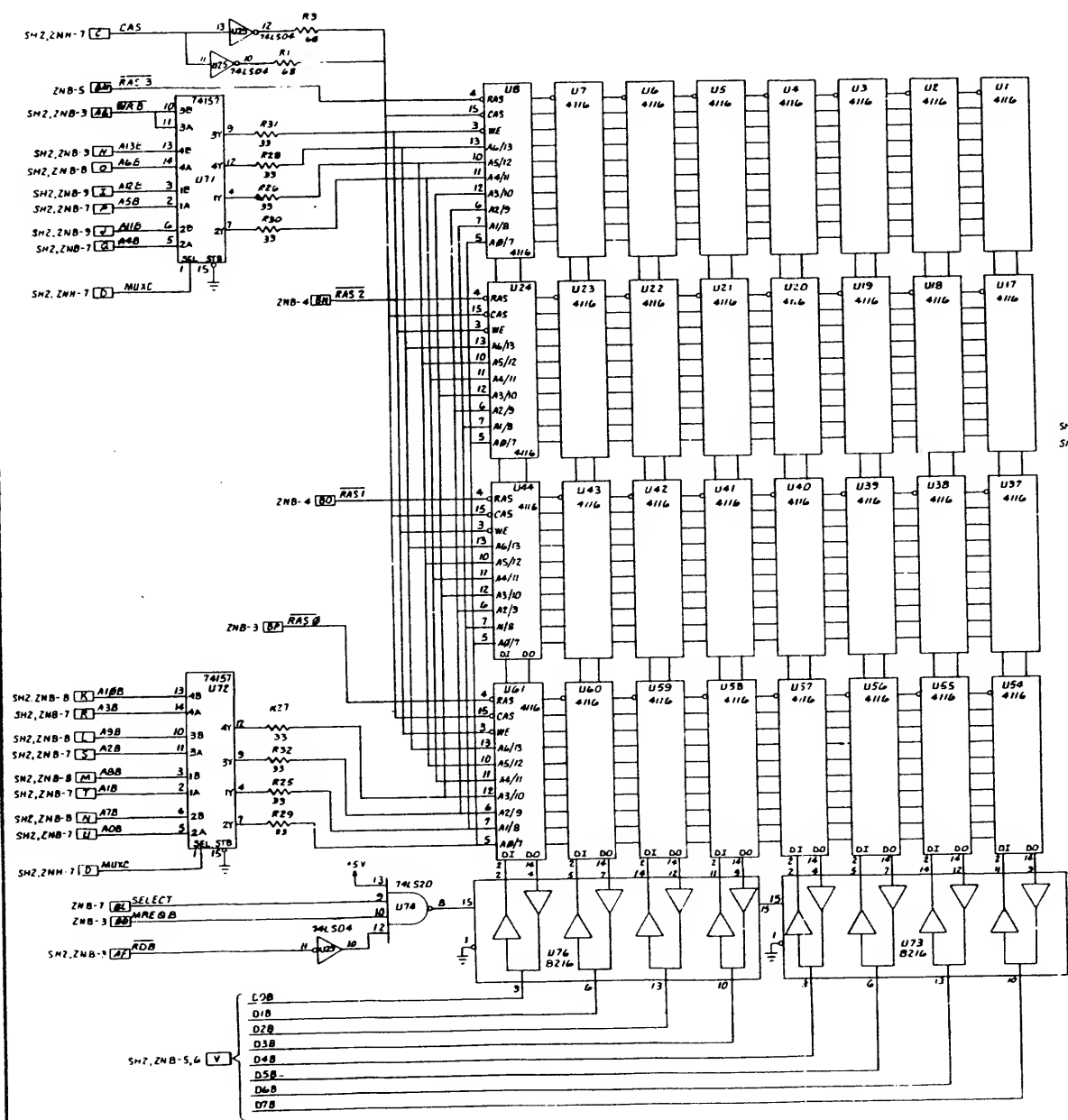


CRT DISPLAY
GENERATOR

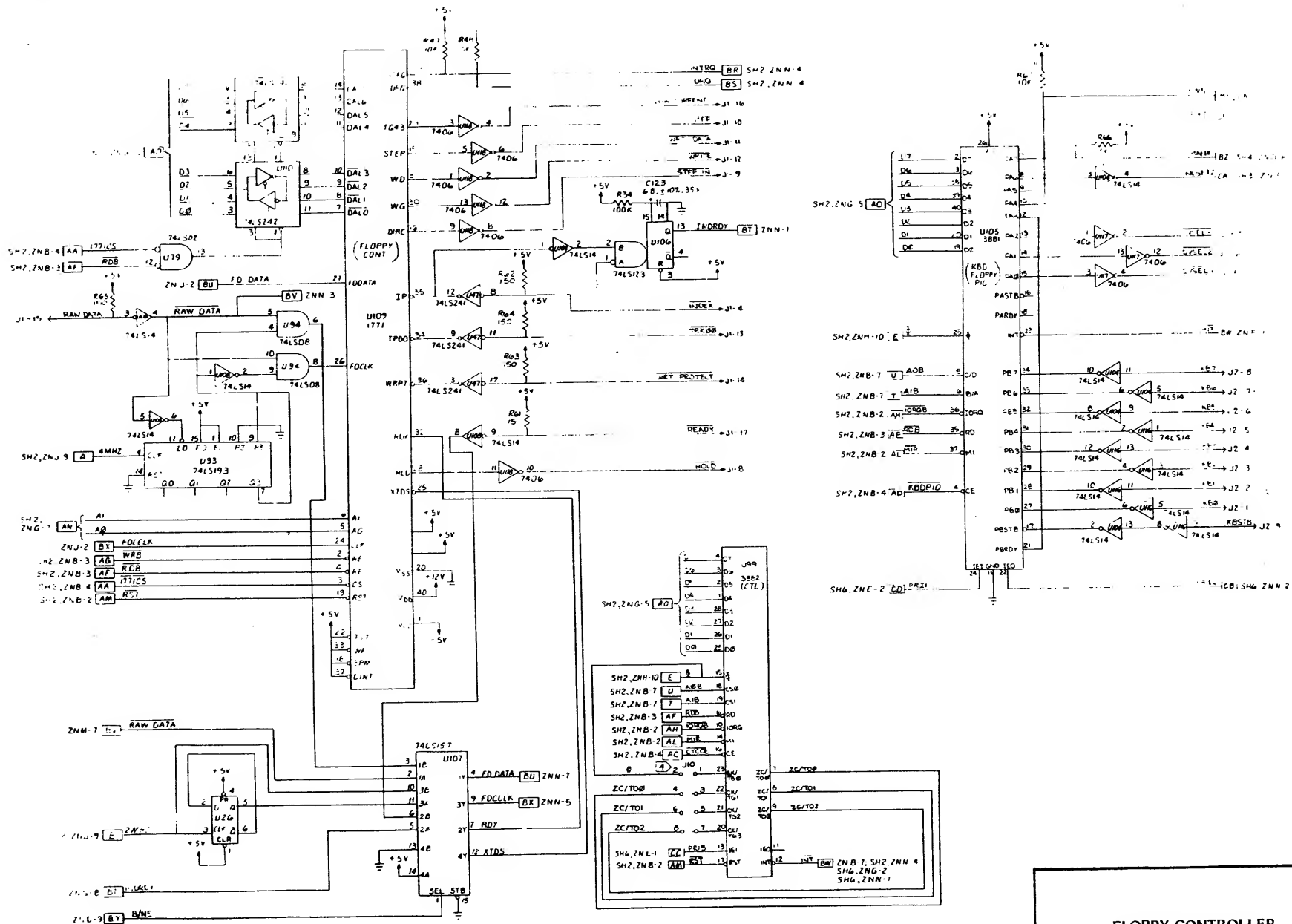
ETCH - 1

Sheet 3 of 6

SCHEMATICS



RAM
ETCH - I Sheet 4 of 6

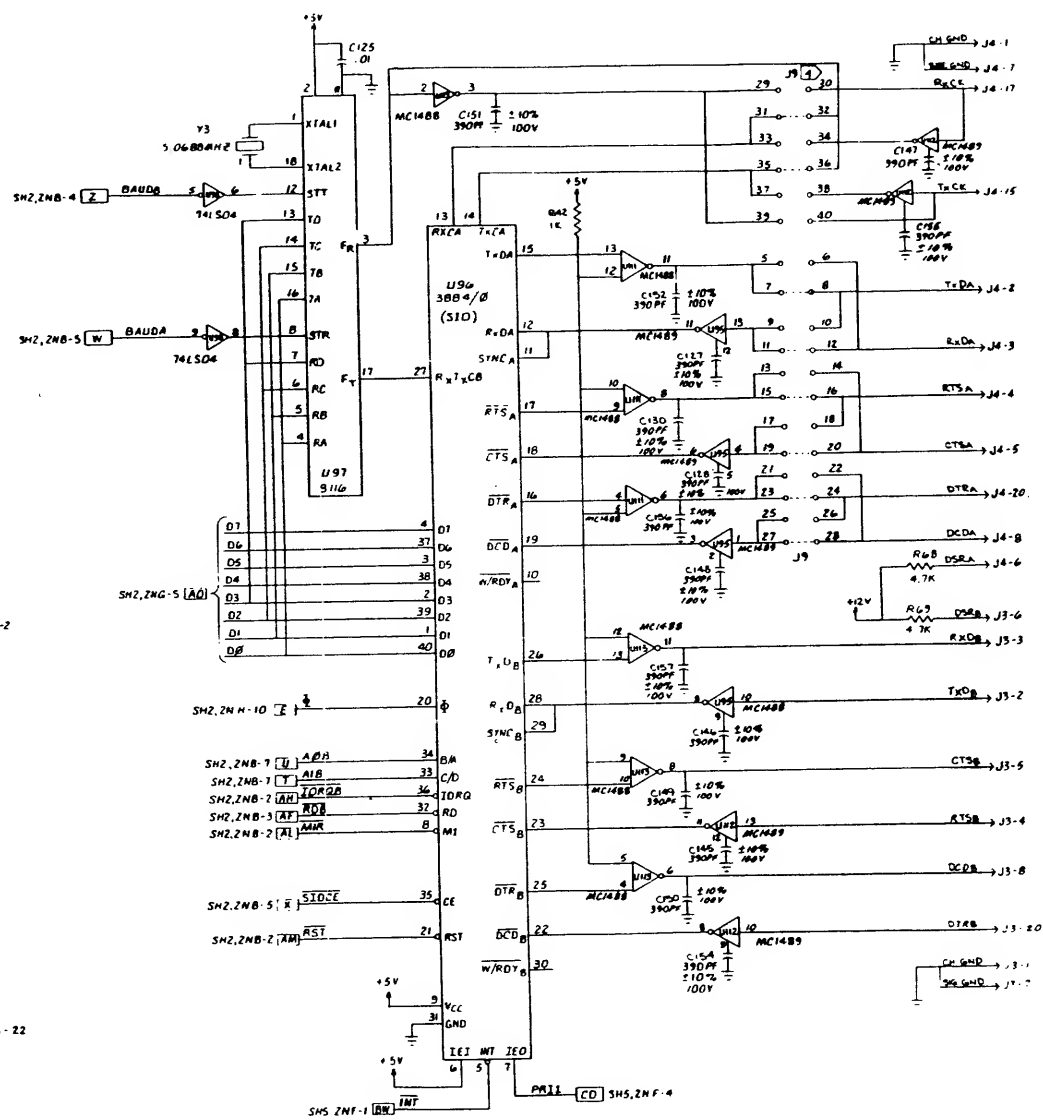


FLOPPY CONTROLLER,
KEYBOARD INPUT, CTC

ETCH - 1

Sheet 5 of 6

SCHEMATICS



GP, PIO, SIO

ETCH - 1

Sheet 6 of 6

SCHEMATICS

CAPACITANCE VALUES ARE IN MICROFARADS, +80 - 20%, 50V

2. POWER DISTRIBUTION TABLE

REF. DESIGNATIONS	GND	+5	+12	-12	-5
U1-8,17-24,37-44 54-61	16	9	8		
U9-12,14-16, 23,26,28 33,34,36,45,49,51,52 53,74,77,78,79,82,83 89,94,95,98,100,102 103,104,108,110,112,114 115-119	7	14			1
U13	12	3			
U27,48,50,62 85-73,75,76,80,88,90 91,93,106,107	8	16			
U35,47,81	10	20			
U46	29	11			
U63,64,92	12	24			
U84-87	9	18			
U96	31	9			
U97	11	2	9		
U99	5	24			
U10X,105	11	26			
U103	20	21	40		1
U11,113	7	14	1		

REFERENCE DESIGNATIONS	
LAST USED	NOT USED
C196	E27, 38, 7, 31, 92, 93, 96
E3	
J11	
R69	R37
S1	
U179	
Q1	
V22	
CR1	

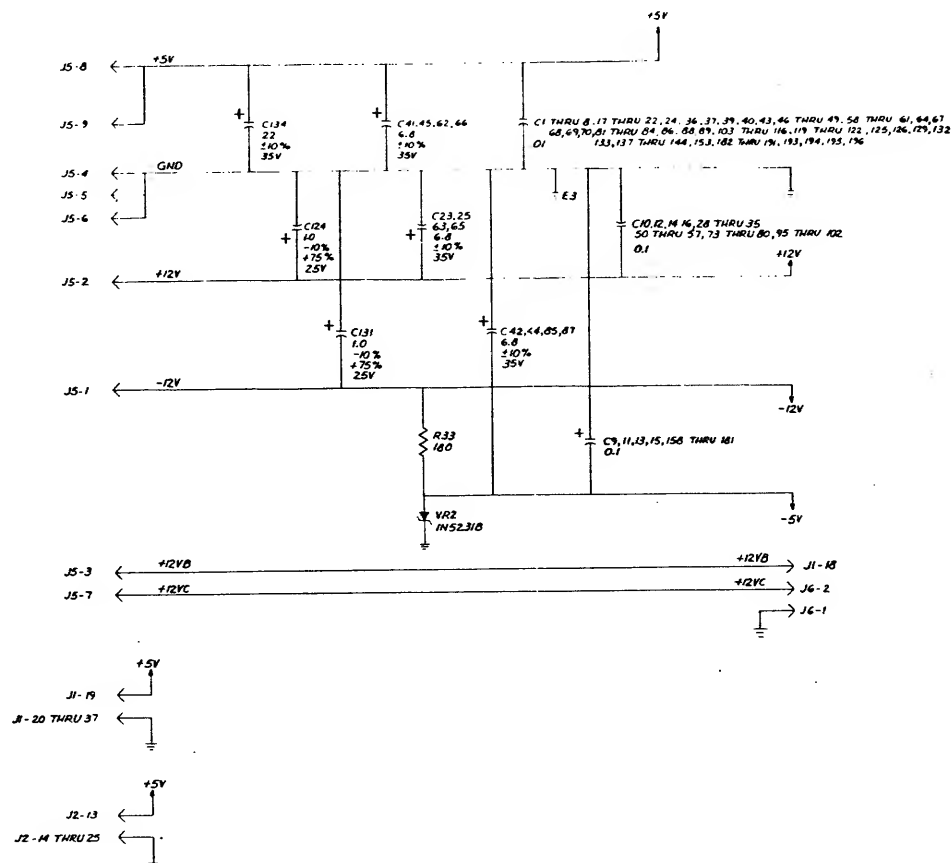
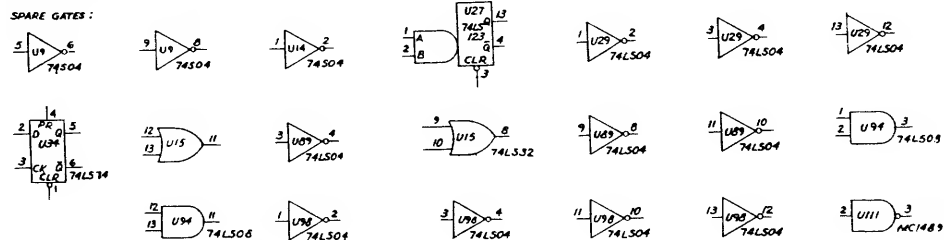
LAST USED	NOT USED
Y3	

4 FOR NORMAL OPERATION SHUNTS TO BE
INSTALLED IN THE FOLLOWING POSITIONS

REF DESIG	BETWEEN PINS
E1	1,2
E2	1,2
J9	7/8, 11/12, 15/16 19/20 23/24 27/28, 31/32 35/36
J10	3/4, 7/8

5. LAST INTERCONNECT LETTER USED "CD"

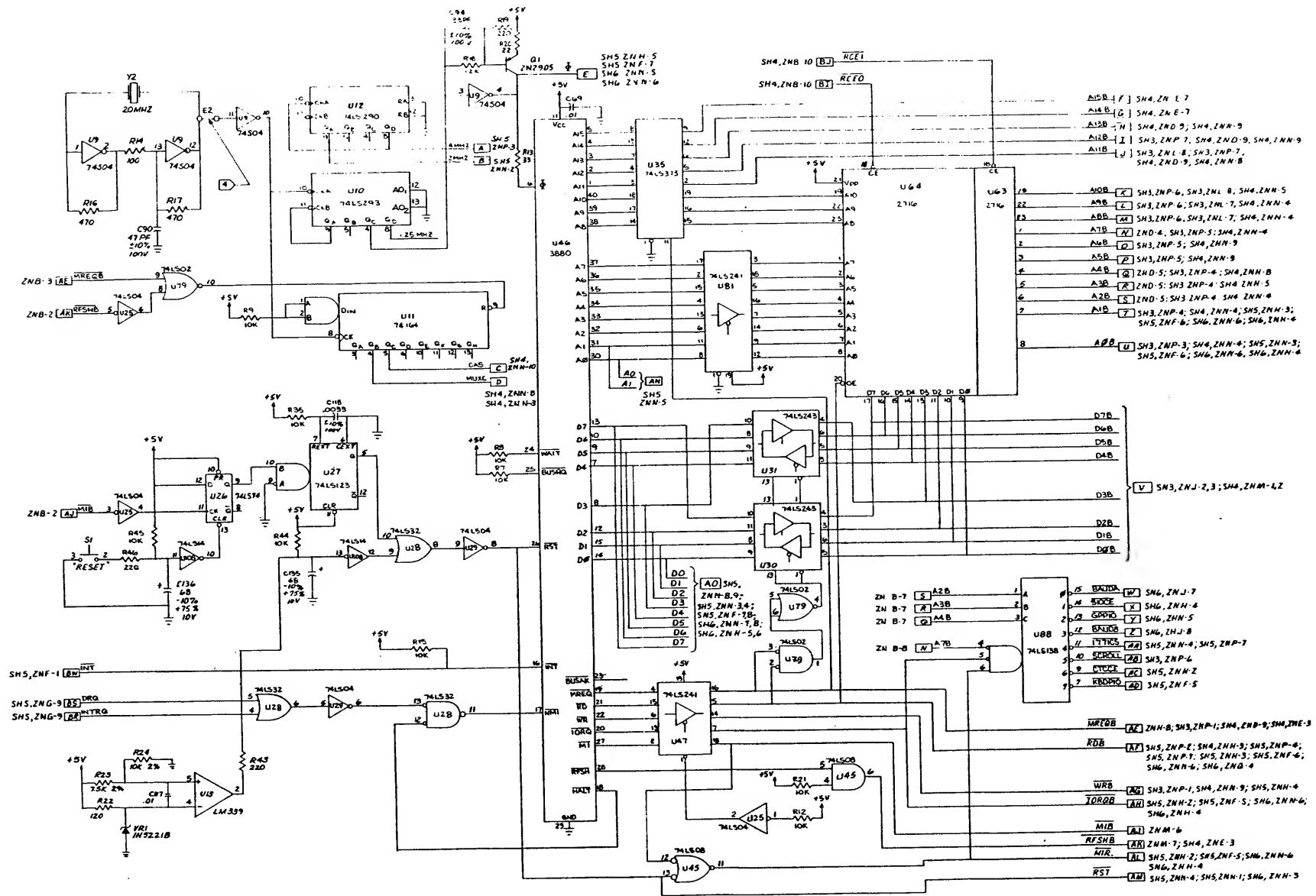
6 SPARE GATES :



POWER DISTRIBUTION

ETCH - 2

Sheet 1 of 6

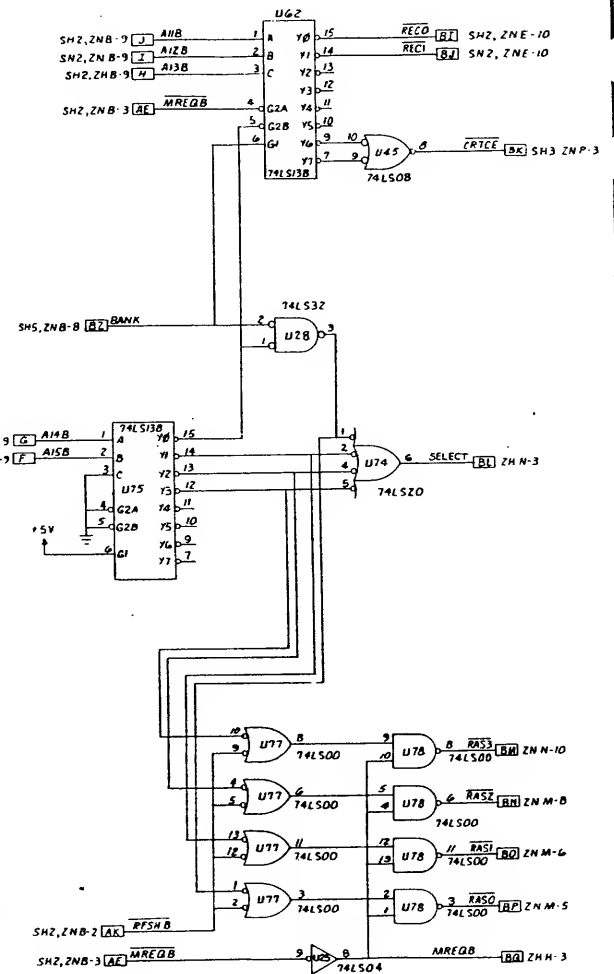
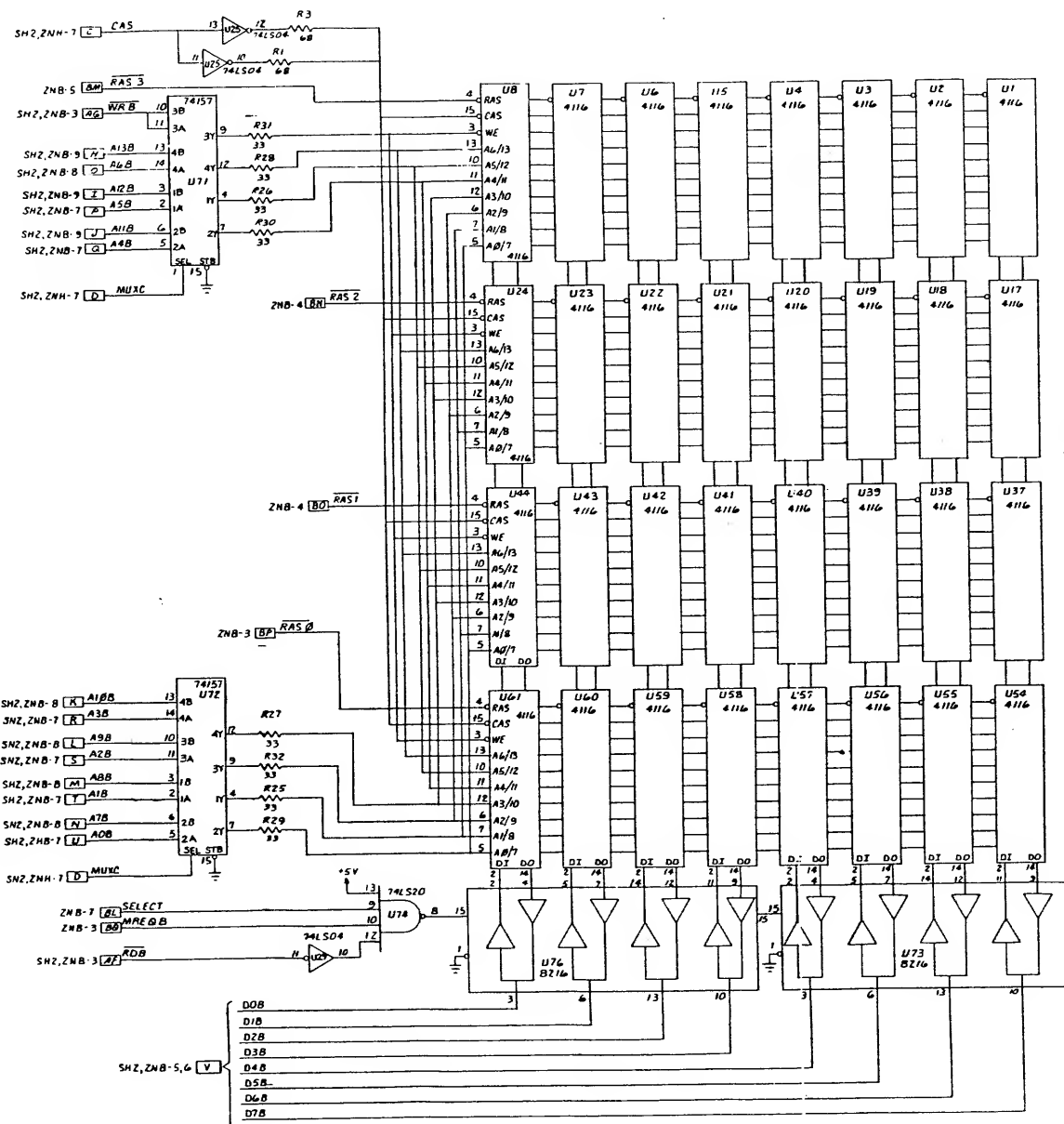


PROCESSOR

ETCH - 2

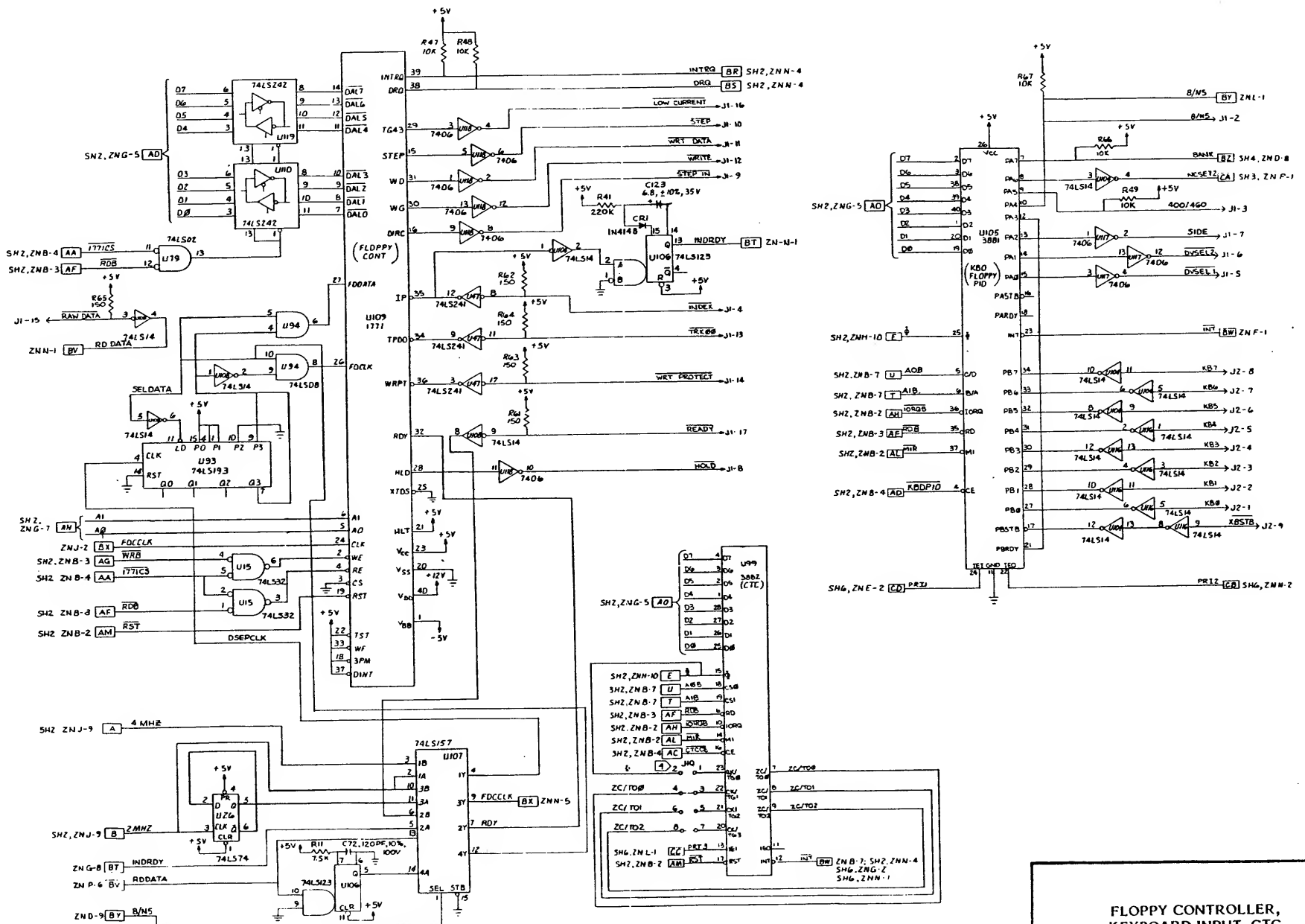
Sheet 2 of 6

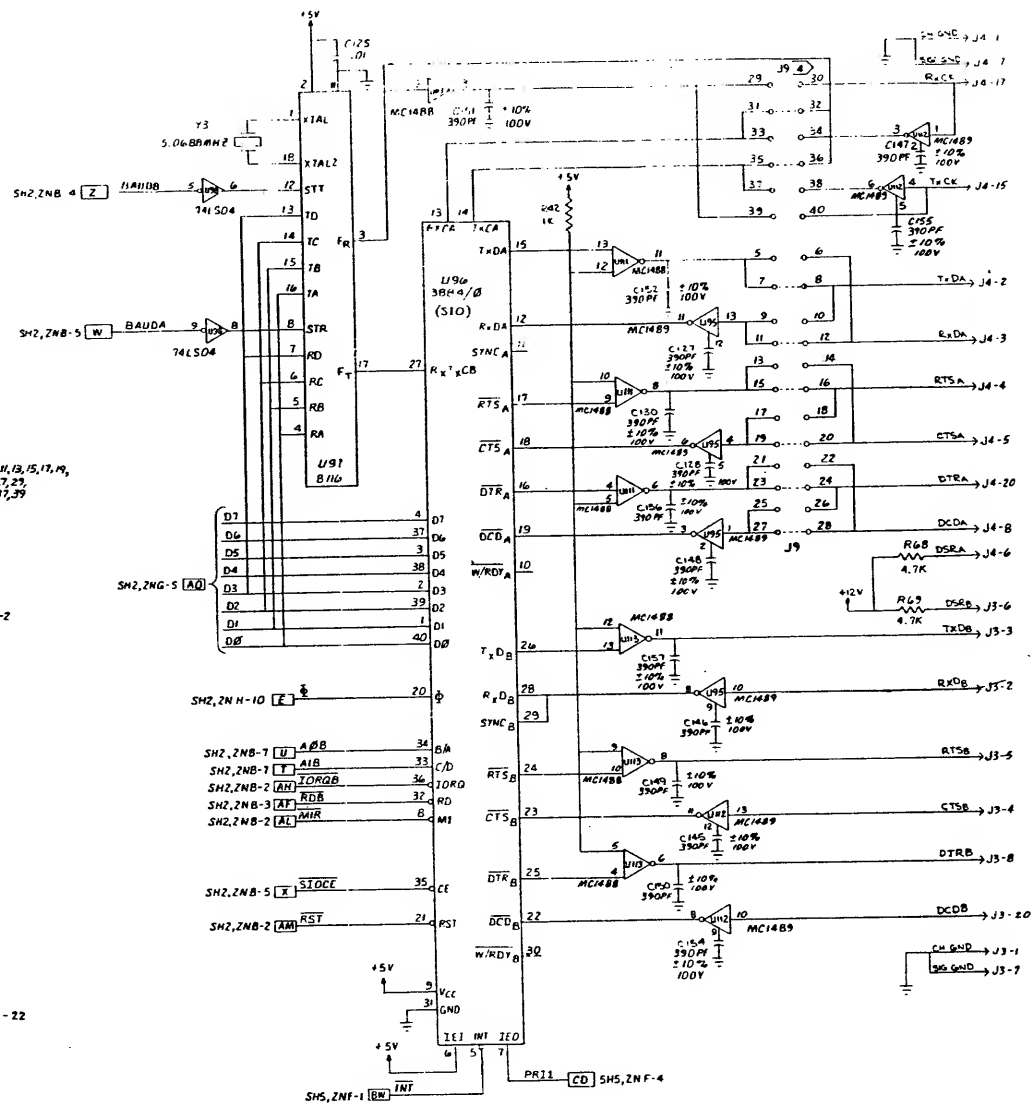
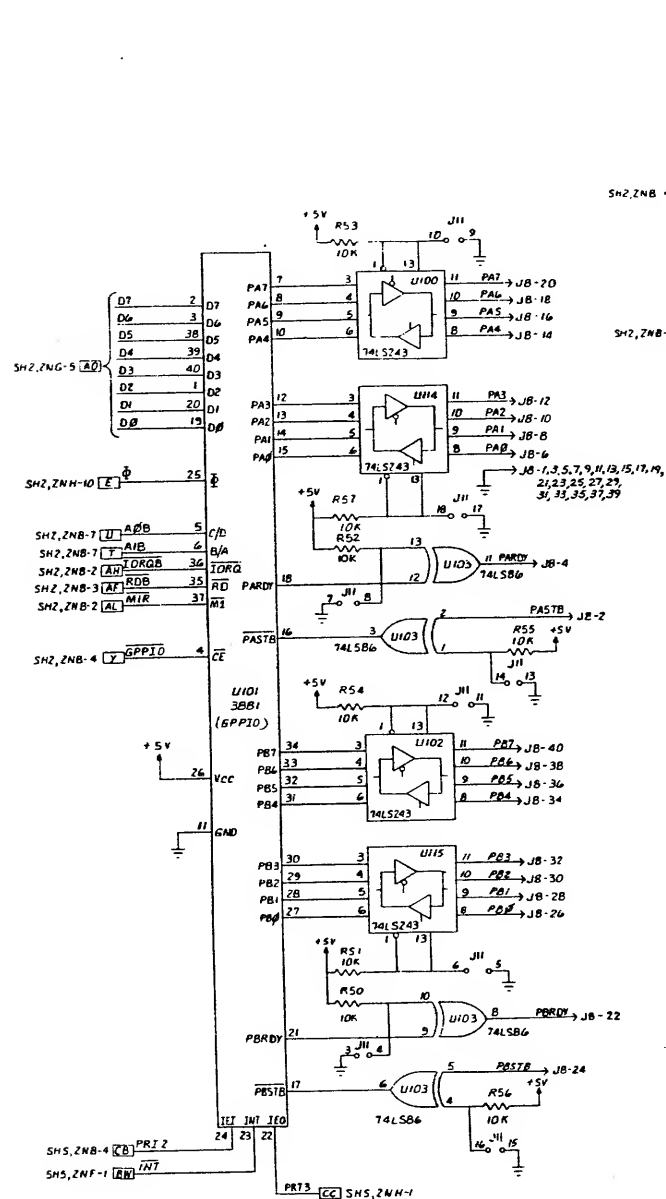
SCHEMATICS



RAM
ETCH - 2 Sheet 4 of 6

SCHEMATICS





GP, PIO, SIO
ETCH - 2 Sheet 6 of 6

SCHEMATICS